File No. S370-34
Order No. GC20-1801-4

Systems

IBM Virtual Machine Facility/370: Planning and System Generation Guide

Release 2 PLC 13

This publication is intended for those responsible for the planning and installation of a VM/370 system. It includes information about:

- Planning for system generation
- Defining your VM/370 system
- Generating VM/370 (CP, CMS, and RSCS)
- Generating a 3704/3705 control program that executes under VM/370
- Generating the standalone 2780 Spool Remote Program
- Updating VM/370

A prerequisite for understanding this publication is the publication *IBM Virtual Machine Facility/370: Introduction,* Order No. GC20-1800.

IBM

Fifth Edition (January 1975)

This is a major revision of GC20-1801-3 and makes that edition and Technical Newsletter GN20-2639, dated June 28, 1974, obsolete. This edition, together with Technical Newsletter GN20-2658, dated March 31, 1975, corresponds to Release 2 PLC 13 (Program Level Change) of IBM Virtual Machine Facility/370 and to all subsequent releases until otherwise indicated in new editions or Technical Newsletters.

Changes are periodically made to the specifications herein; before using this publication in connection with the operation of IBM systems, consult the latest IBM System/360 and System/370 Bibliography, Order No. GA22-6822, and its Virtual Storage Supplement, Order No. GC20-0001, for the editions that are applicable and current.

Technical changes and additions to text and illustrations are indicated by a vertical bar to the left of the change. Because this publication is completely rewritten, you should read the entire publication.

Requests for copies of 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 IBM Corporation, VM/370 Publications, 24 New England Executive Park, Burlington, Massachusetts 01803. Comments become the property of IBM.

© Copyright International Business Machines Corporation 1972, 1973, 1974, 1975

Preface

This publication is intended for system programmers and those responsible for the planning and installation of a VM/370 system. It contains information about VM/370 and the procedures used to generate and support a VM/370 system.

You should have a general understanding of System/370 data processing techniques and be familiar with teleprocessing techniques.

This publication has six Parts, plus Appendixes.

"Part 1: Planning for System Generation" describes the components, features, and options of VM/370, and tells you what you must do during system generation to support them. Part 1 includes information about CMS, RSCS, and other operating systems in a virtual machine. It also discusses performance options, saved systems, minidisks, and lists the devices supported by VM/370.

"Part 2: Defining Your VM/370 System" tells you how to create the files that define your system. The real I/O configuration (DMKRIO), CP system control (DMKSYS), VM/370 Directory (DMKDIR), system name table (DMKSNT), and forms control buffer load (DMKFCB) files, and the macros and control statements needed to create them, are described.

"Part 3: Generating VM/370 (CP, CMS and RSCS)" describes the step-by-step procedure for installing CP, CMS, and optionally RSCS. The three starter systems for CP and CMS (2314, 3330, and 3340) are discussed separately. Also included is a description of the Installation Verification Procedure (IVP) for CP and CMS.

"Part 4: Generating the 3704/3705 Control Program" describes the step-by-step procedure for installing a 3704/3705 control program that runs under VM/370.

"Part 5: The Standalone Spool Remote Program" describes the installation procedure for the 2780 Spool Remote Program (DMKSRP).

"Part 6: Updating VM/370" describes the procedures, programs, and EXEC procedures to update VM/370 source code and macro libraries.

The appendixes include information about:

- The program products supported by CMS
- Configuring VM/370
- Representative VM/370 directory entries
- CMS regeneration requirements
- Notational conventions for commands and macros
- Service programs
- Compatibility between VM/370 and CP-67
- The procedure for generating a 3340 system residence volume from a current 2314/3330 system residence volume and the Program Level Change (PLC) tape
- VM/370 restrictions

An expanded glossary is available in the IBM Virtual Machine Facility/370: Glossary and Master Index, Order No. GC20-1813.

In this publication, the following terms have extended meanings:

- The term 3330 refers to the IBM 3330 Disk Storage, Model 1, 2, or 11, or the IBM 3333 Disk Storage and Control, Model 1 or 11.
- The term 2305 refers to the IBM 2305 Fixed Head Storage, Models 1 and 2.
- The term 3270 refers to the IBM 3275 and 3277 Display Stations.
- The term "typewriter terminal" refers to printer-keyboard devices that produce hard-copy output only (such as the IBM 2741 Communication Terminal, the IBM 3215 Console Printer-Keyboard, or the IBM 3767 Communication Terminal, Model 1 or 2, operating as a 2741).
- The term 2741 refers to the IBM 2741 Communcation Terminal, and also the 3767 Communication Terminal (unless otherwise noted).
- The term "display device" refers to system consoles and terminals that display data on a screen (such as the IBM 3066 System Console or the IBM 3277 Display Station).

- Unless otherwise noted, where the term "Attention key" is used in this publication, the phrase "(or equivalent)" is implied. The equivalent key on the 1050 terminal is the RESET LINE key; on the 3277 terminal, the Enter key. Each of the terminals that can be used with the VM/370 system has a key that is the equivalent of the Attention key on the 2741 (with which you can signal an attention interrupt).

COREQUISITE PUBLICATIONS

IBM Virtual Machine Facility/370:

Command Language Guide for General Users, Order No. GC20-1804

Operator's Guide, Order No. GC20-1806

System Programmer's Guide, Order No. GC20-1807

System Messages, Order No. GC20-1808

Terminal User's Guide, Order No. GC20-1810

Remote Spooling Communications Subsystem (RSCS) User's Guide, Order No. GC20-1816

Release 2 Guide, Order No. GC20-1815

Introduction to the IBM 3704 and 3705 Communications Controllers, Order No. GA27-3051

IBM 3704 and 3705 Communications Controllers, Network Control Program/VS Generation and Utilities Guide and Reference Manual (for OS/VS TCAM Users), Order No. GC30-3007

IBM 3704 and 3705 Communications Controllers, Network Control Program/VS Generation and Utilities, Guide and Reference Manual (for OS/VS and DOS/VS VTAM Users), Order No. GC30-3008

IBM 3704 Control Panel Guide, Order No. GA27-3086

IBM 3705 Control Panel Guide, Order No. GA27-3087

IBM OS/VS Linkage Editor and Loader, Order No. GC26-3813

IBM 2780 Data Transmission Terminal -- Component Description, Order No. GA27-3005

Figure 1 is an overview of the VM/370 library, with the publications grouped according to their probable users.

References in the text to titles of prerequisite and corequisite VM/370 publications will be given in abbreviated form.

Virtual Machine Facility/370 (VM/370) Library

(Release 2 PLC 11)

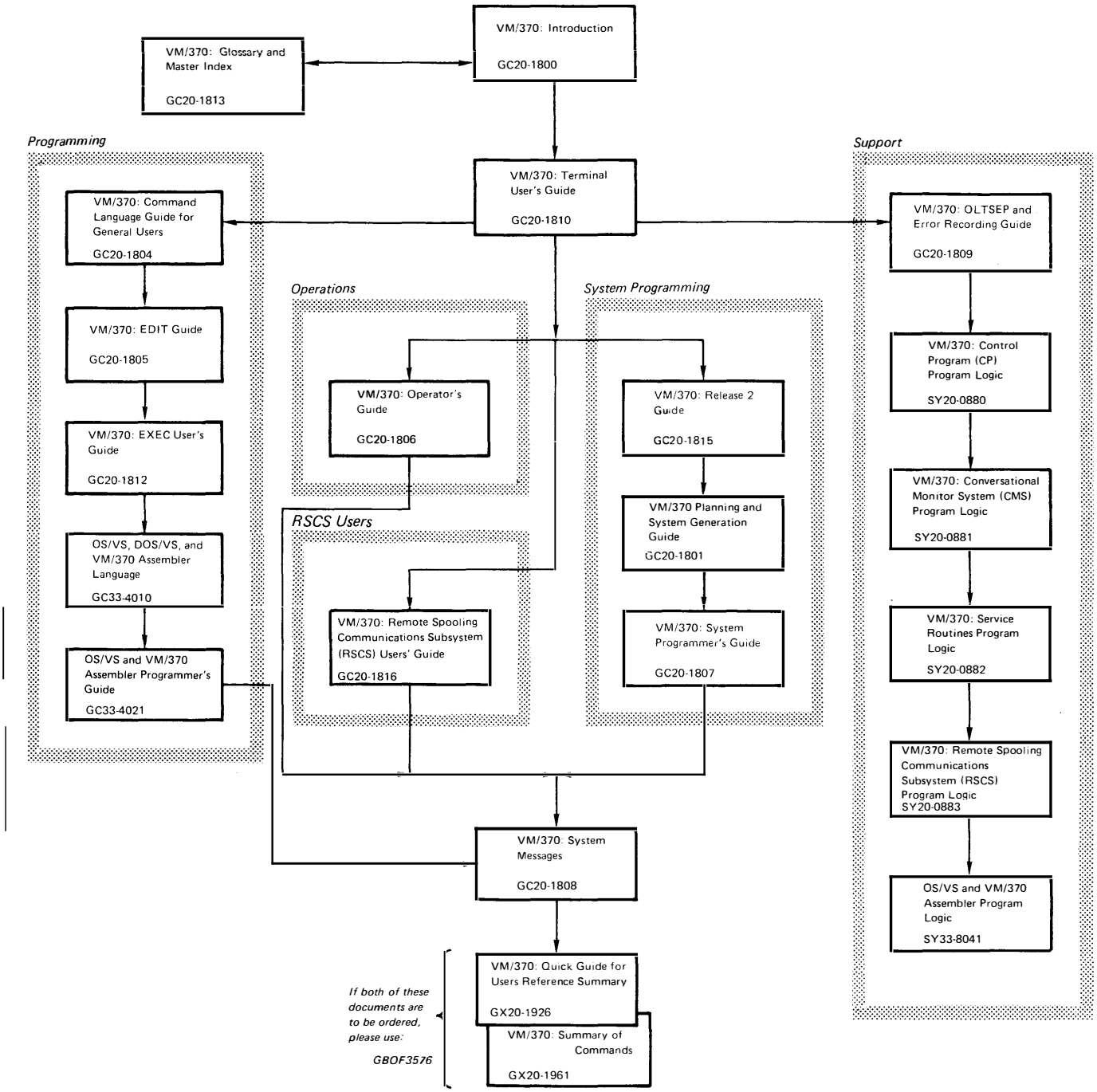


Figure 1. Virtual Machine Facility/370 Library

Summary of Amendments
for GC20-1801-4
as updated by GN20-2658
VM/370 Release 2 PLC 13

VM/370 MEASUREMENT FACILITY

New: Program Feature

The VM/370 control program has two commands that measure system performance: INDICATE and MONITOR. The INDICATE command is new; the function of the MONITOR command is expanded. A new section, "Performance Measurement and Analysis," is added to "Part 1: Planning for System Generation" to reflect this new support.

VM/VS HANDSHAKING FEATURE

New: Program Feature

The VM/VS Handshaking feature is a communication path between VM/370 and OS/VS1 that makes each system control program aware of certain capabilities and requirements of the other. The "Multiprogramming Systems Under VM/370" section of "Part 1: Planning for System Generation" is updated to reflect this new programming feature.

REMOTE 3270 SUPPORT

New: Program Feature

VM/370 now supports the 3270 Display System as a remote virtual machine console attached via nonswitched point-to-point binary synchronous lines to a 2701 Data Adapter Unit or 2703 Transmission Control Unit. Remote 3270s are also supported on 3704/3705 lines in emulation mode. A 3704/3705 is in emulation mode if it is controlled by the Emulation Program (EP) or the Partitioned Emulation Program (PEP) in EP mode.

The following changes to this publication reflect this support:

- The "CMS Planning Considerations" section of "Part 1: Planning for System Generation" is updated.
- A new section, "Planning Considerations for Remote 3270s," is included in "Part 1: Planning for System Generation."
- The "Estimating VM/370 Storage Requirements" section of "Part 1: Planning for System Generation" is updated.
- The "Configurations" section of "Part 1: Planning for System Generation" is updated.
- The "Preparing the Real I/O Configuration File (DMKRIO)" section of "Part 2: Defining Your VM/370 System" is updated. Descriptions of the two new macros, CLUSTER and TERMINAL, are included.
- "Part 4: Generating the 3704/3705 Control Program" is updated.
- "Appendix B: Configuration Aid" is updated.
- "Appendix I: VM/370 Restrictions" contains changes for remote 3270 support.

MISCELLANEOUS CHANGES

Changed: Program and Documentation

Numerous small changes have been made to this publication. In addition, the following changes were made:

- The restriction about using the Virtual Machine Assist Feature with dedicated channels no longer exists. The "Virtual Machine Assist Feature" section of "Part 1: Planning for System Generation" is updated.
- The "Estimating VM/370 Storage Requirements" section of "Part 1: Planning for System Generation" is updated.
- The "Minidisks" section of "Part 1: Planning for System Generation" is updated.

- Corrections have been made to the "Configurations" section of "Part 1: Planning for System Generation".
- Corrections have been made to the "RDEVICE Macro" section of "Part 2: Defining Your VM/370 System."
- The "RCTLUNIT Macro" section of "Part 2: Defining Your VM/370 System" is updated.
- Changes have been made to the OPTION, SPOOL, DEDICATE, and SPECIAL control statements of the Directory program in "Part 2: Defining Your VM/370 System."
- Minor changes have been made to Steps 12 and 17 of the 2314, 3330, and 3340 Starter System descriptions in "Part 3: Generating VM/370 (CP, CMS, and RSCS)."
- "Part 4: Generating the 3704/3705 Control Program" is updated to show the publications needed for Version 3 of the 3704/3705 control program.
- "The Loader" section of "Part 6: Updating VM/370" is updated to show you how to update the loader and create an executable form of the updated loader.
- A new appendix, "Appendix I: VM/370 Restrictions," is included in this publication. This information was previously in the VM/370: Introduction.

PUBLICATION REWRITTEN

New: Documentation

The VM/370: Planning and System Generation Guide is completely rewritten and reorganized. Information that did not pertain to system generation is no longer in this publication. The information in "Section 2. CMS Considerations" and "Section 3. Operating Systems -- General Information" that did not pertain to system generation was moved to the VM/370: Command Language Guide for General Users.

The revised VM/370: Planning and System Generation Guide has six parts and eight appendixes:

- "Part 1: Planning for System Generation"
- "Part 2: Defining Your VM/370 System"
- "Part 3: Generating VM/370 (CP, CMS, and RSCS)"
- "Part 4: Generating the 3704/3705 Control Program"
- "Part 5: The Standalone Spool Remote Program"
- "Part 6: Updating VM/370"
- "Appendix A: IBM Programs Executable Under CMS"
- "Appendix B: Configuration Aid"
- "Appendix C: Representative Directory Entries"
- "Appendix D: CP/CMS Nucleus/Module Regeneration Requirements"
- "Appendix E: Notational Conventions"
- "Appendix F: VM/370 Service Programs Used During System Generation"
- "Appendix G: Compatibility of VM/370 with CP-67/CMS"
- "Appendix H: Creating a 3340 System Residence Volume from a Current 2314 or 3330 System Residence Volume and the PLC Tape"

"Part 1: Planning for System Generation" contains information that was formerly in "Section 1. Introduction," "Section 2. CMS Considerations," "Section 3. Operating Systems -- General Information," and "Section 4. System Generation Planning." The following sections are new:

- "Specifying a Virtual=Real Machine"
- "Virtual Machine Assist Feature"
- "VM/370 Using Channel Switching"
- "Generating a VM/370 System that Supports the 3704/3705"

In addition, system generation considerations are now included in the "CMS Batch Facility" section, and the "Configurations" section is reformatted to improve its readability.

"Part 2: Defining Your VM/370 System" describes the macros and programs that define VM/370. It contains information about the real I/O configuration file, CP system control file, system name table file, VM/370 Directory program, and the Forms Control Buffer. This information was previously in "Section 4. System Generation Planning."

"Part 3: Generating VM/370 (CP, CMS, and RSCS)" describes the step-by-step procedures for generating VM/370. It contains the information that previously was in "Section 5. Generating and Installing VM/370" and "Appendix G: 3330 System Generation."

"Part 4: Generating the 3704/3705 Control Program" describes the step-by-step procedure for installing an Emulation Program, Network Control Program, or Partitioned Emulation Program. Previously, this information was in the VM/370: System Programmer's Guide.

"Part 5: The Standalone Spool Remote Program" describes the procedure for installing the 2780 Spool Remote Program (DMKSRP). Previously, this information was in the VM/370: System Programmer's Guide.

"Part 6: Updating VM/370" tells you how to update VM/370 and describes the programs and EXEC procedures you need. Most of this information was in "Section 6. Supporting and Updating an Installed System." The examples are clarified, the VMFASM command format is added, and the VMFBLD and ASMGEND EXEC procedures are now described.

The following editorial changes were also made in the Appendixes:

- "Appendix C: Command Privilege Classes," "Appendix D: CP Commands," and "Appendix E: CMS Commands" were deleted. This information is in the VM/370: Command Language Guide for General Users.
- "Appendix G: 3330 System Generation" is deleted. The information is included in "Part 3: Generating VM/370 (CP, CMS, and RSCS)."
- "Appendix H: CP Loadlist Requirements" is deleted. The information now is included in the "Using VMFLOAD to Generate a New Nucleus" section of Part 6.

The following Appendixes are new, although their contents are not new:

- "Appendix E: Notational Conventions"
- "Appendix F: VM/370 Service Programs." The service programs previously were described in "Section 4. System Generation Planning."
- "Appendix G: Compatibility of VM/370 with CP-67/CMS." This information was in the VM/370: Command Language Guide for General Users.

NEW DEVICE SUPPORT

New: Program Feature

The IBM 3340 Direct Access Storage Facility is now supported by VM/370. This support includes Rotational Position Sensing (RPS) and the 3348 Data Module, Models 35, 70 and 70F (supported as a Model 70). The following changes to this publication reflect this support:

- The "Introduction to VM/370 System Generation" section of "Part 1: Planning for System Generation" is updated.
- The "CMS Planning Considerations" section of "Part 1: Planning for System Generation" is updated.
- The "Direct Access Storage Requirements for CP" section of "Part 1: Planning for System Generation" is updated.
- The "Estimating DASD Storage Requirements for CMS Minidisks" section of "Part 1: Planning for System Generation" is updated.
- The "Minidisks" section of "Part 1: Planning for System Generation" is updated.
- The "Configurations" section of "Part 1: Planning for System Generation" is updated.
- "Part 2: Defining Your VM/370 System" is updated. The RDEVICE and SYSRES macros are updated and the "Creating Your VM/370 Directory" section is updated.
- "Part 3: Generating VM/370 (CP, CMS, and RSCS) is updated to include a complete description of the system generation procedure for a 3340 starter system.
- "Appendix B: Configuration Aid" is updated.
- "Appendix F: VM/370 Service Programs" is updated.
- "Appendix H: Creating a 3340 System Residence Volume from a Current 2314 or 3330 System Residence Volume and the PLC Tape" is added.

VM/370 supports the IBM 3767 Communications Terminal (at 300 bps) operating as an IBM 2741 Communication Terminal only. The "Configurations" section of "Part 1: Planning for System Generation" is updated to reflect this support.

NEW SUBSYSTEM SUPPORTED AS VM/370 COMPONENT

New: Component

The Remote Spooling Communications Subsystem (RSCS) is now supported as a VM/370 component. RSCS transmits spool files across a teleprocessing network to and from remote stations. Remote stations supported include HASP- and ASP-type batch processors and work stations. The new component is a control program designed to execute in a virtual machine dedicated to remote spooling. The following changes reflect this support:

- "Figure 1. Virtual Machine Facility/370 Library" includes two new publications.
- The "Introduction" section of "Part 1: Planning for System Generation" is updated.
- The "Planning Considerations for Virtual Machine Operating Systems (other than CMS)" section of "Part 1: Planning for System Generation" is updated.
- The "Configurations" section of "Part 1: Planning for System Generation" is updated.
- "Part 3: Generating VM/370 (CP, CMS, and RSCS)" is updated to include a complete description of the system generation procedure for RSCS.
- "Part 6: Updating VM/370" is updated to include information about updating RSCS in the "Building the RSCS Nucleus" section and in "Figure 27. System Support Plan."
- The VMFBLD EXEC procedure is updated to support RSCS. The changes are described in the "Using the VMFBLD EXEC Procedure to Build a New Nucleus" section of "Part 6: Updating VM/370."
- "Appendix C: Representative Directory Entries" has an entry for an RSCS virtual machine.

ENHANCEMENT FOR THE VIRTUAL MACHINE

New: Program Feature

Programs using block multiplexer channel operations (for example, DOS/VS, VS1, and VS2) can now be executed in block multiplexer mode on the virtual machine. This virtual machine option may increase the throughput of the virtual machine's operating system because it allows the overlap of SIOs to virtual devices without encountering the channel busy condition. The following changes to this publication reflect this support:

- The "Performance Guidelines" section of "Part 1: Planning for System Generation" is updated.
- The "Virtual Machine Options" section of "Part 1: Planning for System Generation" is updated to describe the BMX option.

CMS ZAP SERVICE PROGRAM ENHANCED

New: Program Feature

The CMS ZAP service program is enhanced so that it can now modify or dump TXTLIB and MODULE files in addition to LOADLIB files. This support is described in the "Using the ZAP Program to Update CMS MODULE, LOADLIB, or TXTLIB Files" section of "Part 6: Updating VM/370."

READING DOS FILES FROM DOS DISKS

New: Programming Feature

Using OS macros, CMS can now read DOS files that reside on DOS disks. No DOS macros are simulated. The "CMS Planning Considerations" section of "Part 1: Planning for System Generation" is updated.

MISCELLANEOUS

Changed: Program and Documentation

The following miscellaneous changes were made:

- The formula for calculating the maximum size virtual=real area allowed is changed. The formula is described in the "Specify a Virtual=Real Machine" section of "Part 1: Planning for System Generation."
- A new operand for the RDEVICE macro, FEATURE=2CHANSW or =4CHANSW is mentioned in the "VM/370 Using Channel Switching" and "Operating

Systems Using Reserve/Release" sections of "Part 1: Planning for System Generation." The features are described further in the "RDEVICE Macro" section of "Part 2: Defining Your VM/370 System."

- The "Real Storage Requirements for CP" section of "Part 1: Planning for System Generation" is updated.

- The DMKSNT modules that are supplied with the starter system are included in the "Preparing the System Name Table File (DMKSNT)" section of "Part 2: Defining Your VM/370 System."

- The system generation procedure is rewritten. There is a separate procedure for each of the starter systems. Information about installing a CP nucleus with a virtual=real area is added. The system generation procedures are described in "Part 3: Generating VM/370 (CP, CMS, and RSCS)."

- The GENERATE EXEC procedure is changed so that it does not automatically load the CP nucleus if it contains a virtual=real area. This procedure lets you check that there is sufficient virtual storage. Then, if you have enough virtual storage, you can load the CP nucleus from tape. These changes are described in the "Using the GENERATE EXEC Procedure to Generate a New CP or CMS Nucleus" section of "Part 6: Updating VM/370."

- The "RDEVICE Macro" and "RCTLUNIT Macro" sections of "Part 2: Defining Your VM/370 System" are updated to support a new device type, 3158; and a new control unit type, 3158. "Appendix B: Configuration Aid" is also updated.

- The CLASS= operand of the RDEVICE macro is clarified in "Part 2: Defining Your VM/370 System."

- An example of a real I/O configuration file is added to "Part 2: Defining Your VM/370 System."

- The DMKSYS modules that are supplied with the starter systems are added to the "Preparing the CP System Control File (DMKSYS)" section of "Part 2: Defining Your VM/370 System."

- The VM/370 directories that are supplied with the starter systems are added to the "Creating Your VM/370 Directory" section of "Part 2: Defining Your VM/370 System."

- The VMFBLD EXEC procedure is updated to:
  --Not load the CP nucleus if it contains a virtual=real area.

  --Update the CPEREP and Assembler modules and create auxiliary directories for them when it builds the CMS nucleus.

  These changes are described in the "Using the VMFBLD EXEC Procedure to Build a New Nucleus" section of "Part 6: Updating VM/370."

- The CMSGEND EXEC procedure is changed so that it can generate a new Assembler module. This change is described in the "Using The CMSGEND EXEC Procedure to Generate a CMS Module" section of "Part 6: Updating VM/370."

- VM/370 supports the IBM 3704/3705 Network Control Program Support Package for OS/VS (Order No. 5744-BA2) in emulation mode only. "Part 4: Generating the 3704/3705 Control Program" is updated.

IBM 3704/3705 COMMUNICATIONS CONTROLLERS NETWORK CONTROL PROGRAM (NCP) AND PARTITIONED EMULATION PROGRAM (PEP)

New: Program Feature

VM/370 now supports all three of the 3704/3705 control programs:

- Emulation Program (EP)
- Network Control Program (NCP)
- Partitioned Emulation Program (PEP)

The following changes reflect this support:

- The Preface is updated.
- A new section "Generating a VM/370 System that Supports the 3704/3705," is included in "Section 4. System Generation Planning."
- The description of the RDEVICE macro in "Section 4. System Generation Planning" is updated.
- "Appendix B. Configuration Aid" is updated.
- "Figure 32. CP Command Summary" in "Appendix D. CP Commands" is updated.
- "Figure 33. CMS Command Summary" in "Appendix E. CMS Commands" is updated.

MISCELLANEOUS

Changed: Documentation Only

The description of the MACS record in the "Control Files" section of "Section 6. Supporting and Updating an Installed System" is updated.

NEW DEVICE SUPPORT

New: Program Feature

The following devices are now supported:

- IBM 3066 System Console, Model 2
- IBM 3272 Control Unit, Model 2 (local attachment)
- IBM 3277 Display Station, Model 2 (local attachment)
- IBM 3330 Disk Storage, Model 11
- IBM 3333 Disk Storage and Control, Model 11
- IBM 3420 Magnetic Tape Unit, Models 4, 6, and 8 (6250 bpi density)

READ-ONLY OS DATA SET ACCESS SUPPORT

New: Program Feature

CMS can now read, but not write or update, real OS sequential and partitioned data sets.

IBM 3704/3705 COMMUNICATIONS CONTROLLERS

New: Program Feature

This publication now includes information about the IBM 3704 and 3705 Communications Controllers in Network Control Program mode and Partitioned Emulation Program mode as well as in Emulation Program mode.

VIRTUAL MACHINE ASSIST FEATURE

New: Program Feature

The Virtual Machine Assist feature, available on the System/370 Models 135, 145, and 158, reduces VM/370 overhead associated with running OS/VS1 and DOS/VS operating systems in virtual machines. A new option, SVCOFF, has been added to the OPTION statement of the VM/370 Directory service program. See "Directory Creation Service Program (DMKDIR)" in Section 4.

ENHANCEMENTS TO THE CMS EDITOR

New: Program Feature

"Section 2. CMS Considerations" has been changed to reflect enhancements to the CMS Editor associated with support of display devices.

NEW PROGRAM PRODUCTS

New: Programs

The OS PL/I Checkout Compiler, the VS BASIC Processor, and the Planning Systems Generator have been added to the list of Program Products executable under CMS. See "Appendix E. Language Processors and Emulators."

MISCELLANEOUS CHANGES

Changed: Program and Documentation

The DMSGEN and DMKGEN EXEC procedures have been replaced by the VMFMAC EXEC procedure. See "The VMFMAC Procedure" under "Applying Updates" in Section 6.

The GENDECK EXEC procedure has been replaced by the GENERATE EXEC procedure. See "The GENERATE EXEC Procedure" in Section 6.

The system generation procedure has changed. Read "Section 5. Generating and Installing VM/370."

Information on generating a VM/370 system that will include an IBM 3704/3705 Communications Controller has been added. For this purpose, new operands are included in the RDEVICE macro, and a new macro, NAMENCP, has been included. See "Preparing the Real I/O Configuration Deck" and "Configuring the NAMENCP Macro" in Section 4.

The MINIDASD program has been replaced by the OS IBCDASDI service program. See "Virtual Disk Initialization Service Program (IBCDASDI)" in Section 4.

Virtual console spooling now includes CP commands typed in by the user as well as CP messages and responses.

Changed: Documentation only

The section "DMKSRP Installation Procedure," on the IBM 2780 Data Transmission Terminal, has been removed

from this publication and now appears in the VM/370: System Programmer's Guide.

"Section 1. Introduction" now includes an introduction to the system generation process, for novice users.

Information on generating a virtual=real machine now appears under "VM/370 Directory Entries" in Section 4.

A new discussion, "Estimating Storage Requirements for CMS Minidisks," has been added to Section 4.

Information on configuring the NAMESYS macro, which formerly appeared only in the VM/370: System Programmer's Guide, now also appears in Section 4.

Information about the loader has been added to Section 6. See "The Loader (DMKLD00E)."

The discussion "Applying an Update to the System -- An Example" has been replaced by "Recommended Procedures for Local Modification."

"Appendix A. Migration from CP-67/CMS to VM/370" has been removed. The same material is contained in the VM/370: Command Language Guide for General Users in "Appendix G. Compatibility of VM/370 with CP-67/CMS."

"Appendix B. VM/370 Restrictions" has been removed. A complete list of restrictions appears in "Appendix E. VM/370 Restrictions" in the VM/370: Introduction.

The title "Appendix C. CP Command Privilege Classes" has been shortened to "Appendix C. Privilege Classes."

The title "Appendix E. Supported Features and Devices" has been changed to "Appendix A. Language Processors and Emulators."

The new "Appendix B. Configuration Aid" contains charts of control units and associated devices that will be helpful in preparing the Real I/O Configuration Deck.

"Appendix D. CP Commands" and "Appendix E. CMS Commands" (formerly Appendix F) reflect new and changed commands.

In addition, numerous technical corrections, editorial changes, and minor additions have been made throughout this publication.

Contents

PART 1: PLANNING FOR SYSTEM GENERATION . 11

INTRODUCTION 13
Virtual Machine Operating Systems. . . . 13
Introduction to VM/370 System Generation 14

PERFORMANCE GUIDELINES 16
Performance Measurement and Analysis . . 16
Using the Performance Options. 16.1
Specifying a Virtual=Real Machine. . . 16.1
Reserving DASD Space for a CP Nucleus With A Virtual=Real Area. 17
Virtual Storage Requirements 18
Specifying the amount of Virtual=Real Space 18
Virtual Machine Assist Feature 23

CMS PLANNING CONSIDERATIONS. 24
CMS Storage Requirements 24
Virtual Storage. 24
Auxiliary Storage. 24
Device Support 25
CMS Libraries. 26
CMS Command Language 27
CMS Program Language Facilities. 28
CMS Text Processing Facility 28
Limited Support of OS and DOS in CMS . . 29
CMS File Management. 29
CMS Disk Management. 29
File Management. 30
Disk Access. 31
File Sharing in CMS. 32
CMS Tape Support 32
CMS Unit Record Support. 33
Card Reader. 33
Card Punch 33
Printer. 33
Editing. 34
CMS Batch Facility 34
Saving CMS 35

PLANNING CONSIDERATIONS FOR VIRTUAL MACHINE OPERATING SYSTEMS (OTHER THAN CMS). 36
Virtual Machine Options. 37
OS ISAM Channel Programs 38
Multiprogramming Systems Under VM/370. . 38
Multiprogramming Systems without the VM/VS Handshaking Feature 38
The VM/VS Handshaking Feature. 40
Multiple Virtual Machines Using the Same Operating System 40.1
Multiple-Access Virtual Machines . . . 40.2
Performance Considerations 45
The RSCS Virtual Machine 46
RSCS Planning Considerations 46
The ASP Virtual Machine. 47
VM/370 Using Channel Switching 49
Channel Switching Between Two CPUs . . 50
Channel Switching on One CPU 50
Operating Systems Using Reserve/Release. 52

PLANNING CONSIDERATIONS FOR REMOTE 3270S 52.1
3270 Support on Binary Synchronous Lines 52.1
Hardware Configuration Supported . . . 52.2
System Generation Requirements 52.3
The CLUSTER Macro. 52.3
The TERMINAL Macro 52.3
The RDEVICE Macro. 52.3
The Resource Identification Codes. . 52.5
An Example of a Remote 3270 Configuration 52.6

GENERATING A VM/370 SYSTEM THAT SUPPORTS THE 3704/3705 53
Coding the RDEVICE Macro 53
Special Considerations for Coding the RDEVICE Macro 56
Creating an Entry in the System Name Table 59
Reserving DASD Space for the 3704/3705 Control Program Image 59

SAVED SYSTEMS. 60

ESTIMATING VM/370 STORAGE REQUIREMENTS . 61
Real Storage Requirements for CP 61
Reducing the Size of the CP Nucleus. . . 63
Direct Access Storage Requirements for CP. 63
CP Nucleus DASD Requirements 63
Error Recording DASD Requirements. . . 64
Warm Start Data DASD Requirements. . . 64
VM/370 Directory DASD Requirements . . 64
Saved System DASD Requirements 64
Paging and Spooling DASD Requirements. 64.1
Estimating DASD Storage Requirements for CMS Minidisks 65

MINIDISKS. 66
Defining Minidisks 66
Minidisk Space Allocation. 68
Track Characteristics. 68
Alternate Tracks 69
3330 Disks 69
2314, 2319, and 3340 Disks 69
Labels 69
Sharing Minidisks. 70

CONFIGURATIONS 72
VM/370 Minimum Configuration 72
Configurations Supported by CMS. 73
Configurations Supported by RSCS 73
Devices Supported by VM/370. 74
CPUs 75
CPU Required Features and Facilities . 75

Desirable Features 75
Direct Access Storage Devices. 77
Magnetic Tapes 79
Unit Record Devices. 80
Terminals. 81
Special Considerations and Required Features. 81
Transmission Control Units 85
2701 Required Features 85
2702 Required and Optional Features. . 85
2703 Required and Optional Features. . 86
IBM Integrated Communications Attachment (ICA) Required and Optional Features 87
3704/3705 Required Features. 87
Remote Spooling Devices Supported by VM/370. 89
2780 Spool Remote Program (DMKSRP) . . 89
Remote Spooling Communications Subsystem (RSCS). 89
Other Considerations for Planning Your Configuration 92
Two-Channel Switch 92
Devices Used only by an Operating System in a Virtual Machine and Not by VM/370 92

PART 2: DEFINING YOUR VM/370 SYSTEM. . . 93

SYSTEM DEFINITION. 95

PREPARING THE REAL I/O CONFIGURATION FILE (DMKRIO) 96
Coding the Real I/O Configuration Macros for Remote 3270s 96.1
CLUSTER Macro. 96.2
TERMINAL Macro 96.4
RDEVICE Macro. 97
RCTLUNIT Macro 105
RCHANNEL Macro 108
RIOGEN Macro 110
Example of Coding Real I/O Configuration File (DMKRIO) 111

PREPARING THE CP SYSTEM CONTROL FILE (DMKSYS). 114
Performance Considerations for Coding the DMKSYS File Macros. 115
SYSOWN Macro 116
SYSRES Macro 118
SYSOPR Macro 121
SYSCOR Macro 122
SYSTIME Macro. 123
SYSLOCS Macro. 125

CREATING YOUR VM/370 DIRECTORY 126
Considerations for Preparing the Directory Control Statements. 126
System Support Virtual Machines. . . .127
The VM/370 Directory Supplied with the Starter System. 129
2314 Starter System Supplied Directory 130
3330 Starter System Supplied Directory 131
3340 Starter System Supplied Directory 133
Allocating DASD Space for the VM/370 Directory 135
The Directory Program. 136
Invoking the Directory Program (DMKDIR) Under CMS. 137
Invoking Directory as a Standalone Program 138
Directory Control Statements 138

PREPARING THE SYSTEM NAME TABLE FILE (DMKSNT). 151
Coding the NAMESYS Macro 153
Coding the NAMENCP Macro 155

ALTERING THE FORMS CONTROL BUFFER LOAD (DMKFCB). 156

PART 3: GENERATING VM/370 (CP, CMS, AND RSCS) 157

SYSTEM GENERATION PROCEDURES 159
General Information. 160

GENERATING CP AND CMS USING THE 2314 STARTER SYSTEM. 161
Step 1. Load the Format Program from the Starter System Tape 161
Step 2. Format, Label, and Allocate the System Residence Volume 161
Step 3. Label the Starter System Volume.162
Step 4. Load the DASD Dump Restore program from the Starter System Tape. .163
Step 5. Restore the Starter System to Disk. 163
Special Considerations for 800 bpi Tapes 165
Step 6. IPL the Starter System 165
Step 7. Define the Devices Needed to do the System Generation 165
Step 8. Set the Terminal Mode and Spool the Console 167
Step 9. Define or Attach the System Residence Device. 167
Step 10. Make Other Devices Available. .168
Step 11. Load CMS 169
Step 12. Load the PLC (Program Level Change) Tape. 169
Step 13. Punch the Service Programs. . .170
Step 14. Print and Punch the Starter System Supplied Directory, DMKSNT, DMKSYS, and DMKFCB. 171
Step 15. Build the VM/370 Directory and Assemble the Files Defining the Real I/O and System Devices. 172
Format of the READ Control Statement .173
Special Procedure If You Are Using Only One Tape Drive 173
Invoke the GENERATE EXEC Procedure . .173
Step 16. Special Considerations for Virtual=Real Machines 174
Step 17. Load the CP Nucleus 175
Loading a CP Nucleus without a Virtual=Real Area 175
Loading a CP Nucleus that Has a Virtual=Real Area 175

Step 18. IPL the Newly Generated VM/370.176
Step 19. Back up the Newly Generated VM/370. 177
Step 20. Move the CMS System Disk, If You Wish. 178
Step 21. Format the Operator's Virtual 191 Disk. 178
Step 22. Format the MAINT User's Virtual 191 Disk. 179
Step 23. Update CMS. 179
Building a New CMS 180
Step 24. Save CMS. 181

GENERATING CP AND CMS USING THE 3330 STARTER SYSTEM. 182
Step 1. Load the Format Program from the Starter System Tape 182
Step 2. Format, Label, and Allocate the System Residence Volume 182
Step 3. Label the Starter System Volume.183
Step 4. Load the DASD Dump Restore program from the Starter System Tape. .184
Step 5. Restore the Starter System to Disk. 184
Special Considerations for 800 bpi Tapes 185
Step 6. IPL the Starter System 186
Step 7. Define the Devices Needed to do the System Generation 186
Step 8. Set the Terminal Mode and Spool the Console 188
Step 9. Define or Attach the System Residence Device. 188
Step 10. Make Other Devices Available. .189
Step 11. Load CMS. 190
Step 12. Load the PLC (Program Level Change) Tape. 190
Step 13. Punch the Service Programs. . .192
Step 14. Print and Punch the Starter System Supplied Directory, DMKSNT, DMKSYS, and DMKFCB. 192
Step 15. Build the VM/370 Directory and Assemble the Files Defining the Real I/O and System Devices. 193
Format of the READ Control Statement .194
Special Procedure If You Are Using Only One Tape Drive 194
Invoke the GENERATE EXEC Procedure . .194
Step 16. Special Considerations for Virtual=Real Machines 195
Step 17. Load the CP Nucleus 195
Loading a CP Nucleus without a Virtual=Real Area 196
Loading a CP Nucleus that Has a Virtual=Real Area 196
Step 18. IPL the Newly Generated VM/370.197
Step 19. Back Up the Newly Generated VM/370. 198
Step 20. Move the CMS System Disk, If You Wish. 199
Step 21. Format the Operator's Virtual 191 Disk. 199
Step 22. Format the MAINT User's Virtual 191 Disk. 200
Step 23. Update CMS. 200
Building a New CMS 201
Step 24. Save CMS. 202

GENERATING CP AND CMS USING THE 3340 STARTER SYSTEM. 203
Step 1. Load the Format Program from the Starter System Tape 203
Step 2. Format, Label, and Allocate the System Residence Volume 203
Step 3. Label the Starter System Volume.205
Step 4. Load the DASD Dump Restore program from the Starter System Tape. .205
Step 5. Restore the Starter System to Disk. 205
Special Considerations for 800 bpi Tapes 207
Step 6. IPL the Starter System 207
Step 7. Define the Devices Needed to do the System Generation 207
Step 8. Set the Terminal Mode and Spool the Console 209
Step 9. Define or Attach the System Residence Device. 210
Step 10. Make Other Devices Available. .211
Step 11. Load CMS. 211
Step 12. Load the PLC (Program Level Change) Tape. 212
Step 13. Punch the Service Programs. . .213
Step 14. Print and Punch the Starter System Supplied Directory, DMKSNT, DMKSYS, and DMKFCB. 213
Step 15. Build the VM/370 Directory and Assemble the Files Defining the Real I/O and System Devices 214.1
Format of the READ Control Statement .215
Special Procedure If You Are Using Only One Tape Drive 215
Invoke the GENERATE EXEC Procedure . .216
Step 16. Special Considerations for Virtual=Real Machines 216
Step 17. Load the CP Nucleus 217
Loading a CP Nucleus without a Virtual=Real Area 217
Loading a CP Nucleus that Has a Virtual=Real Area 218
Step 18. IPL the Newly Generated VM/370.218
Step 19. Back Up the Newly Generated VM/370. 219
Step 20. Move the CMS System Disk, If You Wish. 220
Step 21. Format the Operator's Virtual 191 Disk. 220
Step 22. Format the MAINT User's 191 Disk. 221
Step 23. Update CMS. 222
Building a New CMS 222
Step 24. Save CMS. 223

VERIFYING CP AND CMS USING THE IVP . . .224
Facilities Required for Each IVP Virtual Machine 224
Starting the IVP 225
Variations of the IVP. 226
Interpreting the Test Results. 227

GENERATING AND INSTALLING RSCS 228
General Information. 228

Defining Your RSCS Virtual Machine . . .228
Defining Your RSCS System. 229
Creating the AXSLINKS COPY File. . . .229
Creating the LAXLINES COPY File. . . .231
Creating the TAGQUEUE COPY File. . . .231
Generation Procedure for RSCS. 231
Step 1. Log On as MAINT and IPL CMS. .232
Step 2. Attach and Load the PLC Tape .232
Step 3. Invoke VMFBLD to Load the RSCS Files. 232
Step 4. Format the RSCS System Disk. .233
Step 5. Create the COPY Files that Define Your RSCS System 233
Step 6. Create DMTLOC Macro Library. .234
Step 7. Assemble the Configuration Table (DMTSYS). 235
Step 8. Invoke VMFBLD to Create the RSCS Nucleus. 235

PART 4: GENERATING THE 3704/3705 CONTROL PROGRAM 237

INTRODUCTION TO THE IBM 3704 and 3705 COMMUNICATIONS CONTROLLERS. 239

PLANNING CONSIDERATIONS. 240
Related Publications 240
3704 and 3705 Support Package. 240
Documentation. 241
VM/370 Support of the 3704 and 3705. . .241
Emulation Program (EP) with VM/370 . .241
Network Control Program (NCP) with VM/370. 242
Partitioned Emulation Program (PEP) with VM/370 243

GENERATING AND LOADING THE 3704/3705 CONTROL PROGRAM 244
Step 1. Log on the VM/370 System 244
Step 2. Set Up a CMS Virtual Machine . .244
Step 3. Load the IBM 3704/3705 Control Program Distribution Tape Files Onto a CMS Disk. 245
Step 4. Code the 3704/3705 Control Program Macro Instructions. 246
BUILD Macro Instruction. 247
CSB Macro Instruction. 248
SYSCNTRL Macro Instruction 248
HOST Macro Instruction 249
The Macro Instructions for the Multiple Terminal Access (MTA) Feature: MTALCST, MTALIST, MTAPOLL, MTATABL, and GROUP. 249
GROUP, LINE, and TERMINAL Macro Instructions. 251
GENEND Macro Instruction 251
Special Macro Coding Considerations for the Partitioned Emulation Program (PEP) 252
Special Macro Coding Considerations for the Emulation Program (EP). . . .252
Step 5. Define the Macro and Text Libraries 252
Step 6. The Stage 1 Generation Procedure 252
The ASM3705 Command. 253
Special Considerations for the Stage 1 Assembly. 254
Step 7. The Stage 2 Generation Procedure 255
The GEN3705 Command. 255
Special Considerations for the Stage 2 Generation Procedure. 257
Step 8. Invoke The EXEC procedure Created in Step 7 257
The LKED Command 258
Step 9. Save the 3704/3705 Control Program Image on Disk 260
The SAVENCP Command. 260
Execution of the SAVENCP Program . . .261
Step 10. Load the 3704/3705 Control Program 262
The NETWORK LOAD Command Line. 262
Execution of the NETWORK LOAD Command.262
Special Considerations for Loading the EP 3704/3705 Control Program. . .263
Special Considerations for Loading the NCP and PEP 3704/3705 Control Programs. 263
Step 11. Logging on Through the 3704/3705 264
Turn the Power On. 264
Check for an Online Message. 264
Follow the Special Sign-on Procedures for 3704/3705 Lines that are in NCP Mode and Also Have the MTA Feature. .265
Logging on After an NCP Control Program Has Abnormally Terminated . .266
Step 12. Applying PTFs to the 3704/3705 Load Library. 266

TESTING THE 3704/3705 CONTROL PROGRAM. .267
NCPDUMP Service Program and How To Use It. 267

PART 5: THE STANDALONE SPOOL REMOTE PROGRAM 269

INTRODUCTION TO IBM 2780 DATA TRANSMISSION TERMINAL 271
IBM 2780 Data Transmission Terminal Characteristics 271

INSTALLING DMKSRP. 272
Step 1. Format Your A-Disk 272
Step 2. Create a DMKSRP ASSEMBLE File. .272
Step 3. Use VMFASM to Assemble DMKSRP. .273
Step 4. Log on the 2780 Virtual Machine.273
Step 5. Punch the DMKSRP Text File . . .274
Step 6. Load the DMKSRP Program. 274
Step 7. Enable the Line. 274
Controlling 2780 Operation 275
Testing 2780 Operation 275

PART 6: UPDATING VM/370. 277

SUPPORTING A VM/370 SYSTEM 279
VM/370 Update Procedures 279
Updating a Module. 280
Control Files. 281
Applying PTFs to VM/370. 283

UPDATING MODULES USING THE VMFASM EXEC PROCEDURE 286
Using VMFASM to Apply IBM-Supplied Updates 287
Using VMFASM to Apply Your Own Updates .288
Other Files Produced by VMFASM 290

USING THE VMFMAC EXEC PROCEDURE TO UPDATE MACRO LIBRARIES. 291

USING VMFLOAD TO GENERATE A NEW NUCLEUS.292
Using VMFLOAD. 292
CP Loadlist Requirements 294

THE LOADER 295
The Load Map 296
Generating a New Loader. 296
Step 1. Assemble the New Loader. . . .296
Step 2. Punch a Copy of the Old Loader. 296.1
Step 3. Punch a Copy of the New Loader Text File. 296.1
Step 4. Load the New Loader. 296.1
Step 5. Punch a Copy of the New Loader (Executable Form). 296.1
Step 6. Name the New Loader DMKLD00E LOADER 296.1

USING THE GENERATE EXEC PROCEDURE TO GENERATE A NEW CP OR CMS NUCLEUS. . . .297

USING THE VMFBLD EXEC PROCEDURE TO BUILD A NEW NUCLEUS 301
Building the CP Nucleus. 301
Special Considerations for Generating a CP Nucleus with a Virtual=Real Area. 303
Building the CMS Nucleus 303
Building the RSCS Nucleus. 304

USING THE CMSGEND EXEC PROCEDURE TO GENERATE A CMS MODULE 306

USING THE ASMGEND EXEC PROCEDURE TO GENERATE THE ASSEMBLER. 308

RECOMMENDED PROCEDURES FOR UPDATING VM/370. 309
Building a CP Nucleus. 311
Building a CMS Nucleus 314
Creating a CMS System Disk 315
Generating the CMS Nucleus 315
Saving the CMS System. 320
CMS Source Programs. 320
Creating CMS Disk-Resident Modules . .321
Using the ZAP Service Program to Update CMS MODULE, LOADLIB, or TXTLIB Files. .322
ZAP Input Control Records. 323
Special Considerations For Using The ZAP Service Program 329
Building an RSCS Nucleus 329
Creating an RSCS System Disk 329
Generating the RSCS Nucleus. 330

APPENDIXES 333

APPENDIX A: LANGUAGE PROCESSORS AND EMULATORS 335
VM/370 Assembler 335
Program Products 335
Installed User Programs. 335
Integrated Emulators 336

APPENDIX B: CONFIGURATION AID. 337

APPENDIX C: REPRESENTATIVE DIRECTORY ENTRIES 341
An Operator's Virtual Machine (OPERATOR). 341
A Virtual Machine that Receives System Dumps (OPERATNS). 341
A Virtual Machine for Updating and Supporting VM/370 (MAINT) 341
A Hardware Service Virtual Machine (SRMAINT) 342
CMS Virtual Machines (USER1, USER2, and USER3). 342
A Batch Virtual Machine (BATCH). 342
A VM/370 Virtual Machine (TESTSYS) . . .342
The Remote Spooling Communications Subsystem (RSCS) Virtual Machine. . . .342

APPENDIX D: CP/CMS NUCLEUS/MODULE REGENERATION REQUIREMENTS 347

APPENDIX E: NOTATIONAL CONVENTIONS . . .349

APPENDIX F: VM/370 SERVICE PROGRAMS USED DURING SYSTEM GENERATION 353
DASD Dump Restore (DDR) Service Program and How To Use It 354
Invoking DDR under CMS 354
Invoking DDR as a Standalone Program .354
DDR Control Statements 355
I/O Definition Statements. 355
Disk Backup Via DASD Dump Restore Service Program (DMKDDR). 363
Examples 363
IBCDASDI -- General Information. 366
Initializing a Minidisk. 366
Assigning an Alternate Track 372
Invoking the IBCDASDI Program. 374
Formatting Volumes--General Information.375
Format/Allocate Service Program (DMKFMT). 377
Format/Allocate Program Card Input . .377
Format/Allocate Console Input. 380

APPENDIX G: COMPATIBILITY OF VM/370 WITH CP-67/CMS. 383
VM/370 CMS Support of CP-67/CMS Commands. 389
CP-67/CMS Macros and Functions and Corresponding CMS Macros. 397

APPENDIX H. CREATING A 3340 SYSTEM RESIDENCE VOLUME FROM A CURRENT 2314 OR 3330 SYSTEM RESIDENCE VOLUME AND THE PLC TAPE. 399
Step 1. IPL CMS. 399
Step 2. Load New CP Macro Library. . . .399

Step 3. Update the Real I/O Configuration (DMKRIO) File 400
Step 4. Update CP and CMS with the PLC Changes 400
Step 5. Format, Label, and Allocate the 3340 Volumes. 401
Step 6. Update the Current VM/370 Directory 402
Step 7. Format Minidisk Areas. 403
Step 8. Copy the Minidisks 404
Step 9. Update and Assemble the CP System Control (DMKSYS) and System Name Table (DMKSNT) Files 404
Step 10. Update the VM/370 Directory and Load It on a 3340 Volume. 404
Step 11. Generate a New CP Nucleus on a 3340 Volume 405
Step 12. Generate a New CMS Nucleus on a 3340 Volume 405
Step 13. Change the Label of the 3340 Disk Labeled TCPVOL 406

APPENDIX I: VM/370 RESTRICTIONS. . . .406.1
Dynamically Modified Channel Programs.406.1
Minidisk Restrictions. 406.1
Timing Dependencies. 406.2
CPU Model-Dependent Functions. 406.3
Virtual Machine Characteristics. . . .406.3
CMS Restrictions 406.6
Miscellaneous Restrictions 406.7

INDEX. 407

Figures

Figure 1. Virtual Machine Facility/370 Library.....9
Figure 2. Devices Supported by a CMS Virtual Machine.....25
Figure 3. File Format of Logical Records.....30
Figure 4. CP and CMS Disk Access.....31
Figure 5. Virtual Devices: 3270 Terminals.....40.2
Figure 6. Virtual Devices: Teleprocessing.....41
Figure 7. A Virtual Multiple-Access System.....42
Figure 8. VM/370 Directory Entry for an APL\DOS-360 Virtual Machine.....43
Figure 9. A Communications System Test.44
Figure 10. A Virtual TCU Running Remote 3270 Type Units.....45
Figure 11. A Remote Spooling Communications Subsystem.....47
Figure 12. A Virtual Machine ASP System.48
Figure 13. A Virtual Machine in an ASP Configuration.....49
Figure 13.1 Example of Determining Line Code for Remote 3270 Resource Identification Codes.....52.6
Figure 13.2 Sample List of Resource Identification Codes for Operations.....52.8
Figure 14. IBM 3704/3705 Models.....55
Figure 15. Use and Definition of Minidisks.....67
Figure 15.1 Remote 3270 Control Unit and Device Addressing.....96.7
Figure 15.2 Examples of Remote 3270 Addressing.....96.8
Figure 16. Example of a Real Configuration.....112
Figure 17. Format of a 2314 Restored Disk.....164
Figure 18. Allocation of the VM/370 Starter System Volume (CPV2L0) when the 2314 Starter System Directory Is Used.....177
Figure 19. Format of a 3330 Restored Disk.....185
Figure 20. Allocation of the Starter System Volume (CPV2L0) when the 3330 Starter System Directory Is Used....198
Figure 21. Format of a 3340 Restored Disk.....206
Figure 22. Allocation of the Starter System Volume (CPV2L0) when the 3340 Starter System Directory Is Used....219
Figure 23. BUILD Macro Operands for VM/370.....247
Figure 24. LCTYPE Operands for MTA Macros with VM/370.....249
Figure 25. Example of MTA Macros for Use with VM/370.....250
Figure 26. Example of Configuration Macros for Use with VM/370..251
Figure 27. System Support Plan.....285
Figure 28. Files for a System Update...310
Figure 29. IBM Program Products.....335
Figure 30. Integrated Emulators that Execute under VM/370.....336
Figure 31. Annotated Sample of Output from the TYPE and PRINT Functions of the DDR Program.....362
Figure 32. Format of 3330 Cylinders for use by CP.....375
Figure 33. 3330 Cylinder 0 Format.....376
Figure 34. Using the Format Program Label Function.....381
Figure 35. Using the Format Program Allocate Function.....381
Figure 36. Using the Format Program Allocate Overlap Function...382
Figure 37. VM/370 Compatibility with CP-67.....384

Part 1: Planning for System Generation

Part 1 contains planning information. It describes the various components, options, and features of VM/370 and tells you what you must do to install them. Part 1 contains the following sections:

- Introduction
- Performance Guidelines
- CMS Planning Considerations
- Planning Considerations for Virtual Machine Operating Systems (Other than CMS)
- Generating a VM/370 System that Supports the 3704/3705
- Saved Systems
- Estimating VM/370 Storage Requirements
- Minidisks
- Configurations

Introduction

The IBM Virtual Machine Facility/370 is System Control Program (SCP) that manages a real computing system so that all its resources -- CPU, storage, and input/output devices -- are available to many users at the same time. Each user has at his disposal the functional equivalent of a real, dedicated computing system. Because this functional equivalent is simulated by VM/370 and does not really exist, it is called a "virtual" machine.

VM/370 supports IBM System/370 Models 135, 145, 155 II, 158, 158MP (uniprocessor mode), 165 II, 168, and 168MP (in uniprocessor mode). The real System/370 must have the Dynamic Address Translation feature, a hardware facility that translates virtual storage addresses to real storage addresses, and the System Timing facility. Also, it must operate in extended control mode, a mode in which all the features of a System/370, including dynamic address translation, are operational.

VM/370 has three components:

- The control program (CP), which controls the resources of the real computer to provide multiple virtual machines.
- The Conversational Monitor System (CMS), which provides a wide range of conversational and time-sharing facilities. Using CMS, you can create and manage files, and compile, test, and execute problem programs.
- The Remote Spooling Communications Subsystem (RSCS), which transfers spool files between VM/370 users and remote locations over telecommunication lines.

For more information about VM/370, see the VM/370: Introduction.

VIRTUAL MACHINE OPERATING SYSTEMS

While the control program of VM/370 manages the concurrent execution of the virtual machines, it is also necessary to have an operating system managing the work flow within each virtual machine. Because each virtual machine executes independently of other virtual machines, each one may use a different operating system, or different releases of the same operating system.

The operating systems that can run in virtual machines are:

Batch or Single User Interactive	Multiple-Access
DOS	APL\DOS-360
DOS/VS	(with CP option)
OS/PCP	VM/370
OS/MFT	Time Sharing
OS/MVT	Option of OS
OS/VS1	
OS/VS2	Conversational
OS-ASP	CMS
RSCS	
PS44	

CP provides each of these with virtual device support and virtual storage. The operating systems themselves execute as though they were controlling real devices and real storage, but they must not violate any of the restrictions listed in the VM/370: Introduction.

INTRODUCTION TO VM/370 SYSTEM GENERATION

The purpose of the system generation is to create a system that meets your installation's particular needs.

The first step in the system generation procedure is to restore the starter system, a small working copy of a basic VM/370 system. Using the starter system, you tailor a VM/370 system to your own hardware configuration. You also describe your DASD volumes and define how they are to be used.

There are three versions of the starter system you can order:

- 2314 Starter System
- 3330 Starter System
- 3340 Starter System

The 2314 starter system must be restored to a 2314 disk, but you can use it to build a 2314 or 3330 system residence volume. The 3330 starter system must be restored to a 3330 disk, but you can use it to build a 2314 or 3330 system residence volume. The 3340 starter system must be restored to a 3340 disk, but you can use it to build a 2314, 3330, or 3340 system residence volume.

Before you begin the system generation procedure, you should:

- Know which devices to include in your VM/370 system.
- Create the real I/O configuration (DMKRIO) file describing your I/O configuration.
- Decide how many virtual machines to define.

- Create the VM/370 directory control statement file describing the virtual machines.
- Decide which volumes are to be owned and used by CP (for system residence, paging, spooling, and so on), the amount of real storage available to VM/370, and the user identification of the real system operator.
- Create the CP system control (DMKSYS) file describing CP-owned volumes, the real storage size, and so on.
- If you wish, you can create your own forms control buffer (module DMKFCB) and system name table (module DMKSNT). These modules are, however, supplied with the starter system.

Once you have defined your VM/370 system with these files, you can begin the system generation procedure. You should read the rest of Part 1 to be sure you have all the information you need to generate your system. Part 2 has the information you need to code the files that define your system and Part 3 describes the system generation procedure step-by-step. Before you start the system generation procedure be sure you have the following manuals available:

VM/370: Command Language Guide for General Users

VM/370: Operator's Guide

VM/370: System Messages

VM/370: Terminal User's Guide

During the system generation procedure, you apply the PLC tape supplied with the starter system. This updates your system to the current level. Then use the Installation Verification Procedure (IVP) to verify that the VM/370 system is functioning properly.

If you wish to install the Remote Spooling Communications Subsystem (RSCS), do so after running the IVP. After you generate VM/370 (CP, CMS, and optionally RSCS) you can generate the 3704/3705 Control Program or the 2780 Spool Remote Program. Information about generating these two programs is found in Parts 4 and 5, respectively, of this manual.

You can generate a 3340 system residence volume without using the 3340 starter system if you have:

- A Release 2 2314 or 3330 system residence volume
- The PLC tape that includes the 3340 support

"Appendix H. Creating a 3340 System Residence Volume from a Current 2314 or 3330 System Residence Volume and the PLC Tape" is a description of the procedure.

Note: VM/370 strongly recommends that you use the 3340 starter system to create a 3340 system residence volume. The procedure described in Appendix H is complex and restricting; it requires that you generate CP and CMS twice.

Performance Guidelines

The performance characteristics of an operating system when it is run in a virtual machine environment are difficult to predict. This unpredictability is a result of several factors:

- The System/370 model used
- The total number of virtual machines executing
- The type of work being done by each virtual machine
- The speed, capacity, and number of the paging devices
- The amount of real storage available
- The degree of channel and control unit contention, as well as arm contention, affecting the paging device
- The type and number of VM/370 performance options in use by one or more virtual machines

Also, the virtual machine's channel mode, block multiplexer or selector, has an effect on the virtual machine's performance.

PERFORMANCE MEASUREMENT AND ANALYSIS

The VM/370 control program has two commands that measure system performance and thus help you identify problem areas. The MONITOR command collects system measurement data offline for the system operator or system analyst, while the INDICATE command collects system measurement data online for the system analyst or general user.

The MONITOR command gathers data relating to most aspects of system performance and writes the data on tape. Using the operands of the MONITOR command, you can designate the type of information you want collected. You must summarize or analyze the data written on tape to determine performance trends. You can write your own program to analyze the data collected on tape or use an Installed User Program (IUP) which is available through IBM. When the data collected on tape is summarized, it may indicate the conditions contributing to performance degradation.

The INDICATE command displays, at the terminal, some key information about the system which shows the current performance indicators. INDICATE displays the system conditions existing at the time the command is issued. If, after using the INDICATE command, the system analyst wants more extensive data collection and reduction, he can use the MONITOR command.

For information about using the MONITOR and INDICATE commands and for the format of the tape records created by the MONITOR command, see the VM/370: System Programmer's Guide.

USING THE PERFORMANCE OPTIONS

The performance of a specific virtual machine can be improved by assigning it one or more performance options. These include: favored execution, priority, reserved page frames, locked pages, and virtual=real.

The performance of a VM/370 system running virtual storage operating systems can be improved if you use the virtual machine assist feature. This feature is available as a hardware feature on the System/370 Models 135, 145, and 158 and as an RPQ on the Model 168. In order to invoke the favored execution, priority, reserved page frames, and locked pages options you must have a virtual machine defined with the appropriate command privilege classes. Usually, the operator's virtual machine has the appropriate command classes. Additional planning is needed to support the virtual=real option and virtual machine assist feature. All of these performance options are described in detail in the VM/370: System Programmer's Guide.

SPECIFYING A VIRTUAL=REAL MACHINE

Although the virtual=real option eliminates paging, its main function is to bypass CCW translation. This is possible because I/O from a virtual machine occupying a virtual=real space contains a list of CCWs whose data addresses reflect the real storage addresses.

The only exception is virtual page 0. Virtual page 0 does not exist as real page 0; it is relocated beyond the highest page of the virtual=real area. In order for the virtual machine to perform I/O into virtual page 0, the CCW addresses must be translated.

When CP loads an operating system into a virtual=real area, it turns on CCW translation. Once the operating system is loaded, the operator of the virtual machine should issue a CP command to turn CCW translation off.

When the virtual machine is operating with CCW translation off, it must not perform I/O into virtual page 0. Most operating systems can be generated so that they do not use this area for input/output. However, violdation of this restriction may cause damage to the other virtual machines.

The size of the virtual=real area is specified during CP system generation. It must be large enough to contain the entire addressing space of the largest virtual machine that you execute in the virtual=real area.

Only one virtual=real area can be defined.

Only one virtual machine at a time can occupy the virtual=real area.

Since the virtual=real option removes pages from the dynamic paging area, it affects the performance of other virtual machines.

The virtual=real area is set up at VM/370 initial program load (IPL). It can be released by the primary system operator to be used as part of the dynamic paging area. Once released, it cannot be reclaimed except by reloading VM/370. The virtual=real area must be released in total, that is, unused pages of the area cannot be selected for release.

To use the virtual=real option effectively on a multiport teleprocessing system with no CCW translation (SET NOTRANS ON), lines must be dedicated to that system via the ATTACH command or by VM/370 directory assignment. Conversely, on a multiport teleprocessing virtual=real operation, virtual 2701/2702/2703 lines, (that is, lines assigned and used by CP's DEFINE and DIAL commands) operate with CCW translation. If you invoke the SET NOTRANS ON command it is nullified when a valid DIAL command is detected.

Note that you cannot execute programs with dynamically or self-modifying channel programs in a virtual=real area if you also use the DIAL command.

To generate CP so that it properly supports a virtual=real area you must do the following:

- Specify the VIRT=REAL option in the VM/370 directory for all the virtual machines in your installation that you plan to run in the virtual=real area.
- Reserve enough DASD space for the CP nucleus. A CP nucleus that supports a virtual=real area is larger than one that does not.
- Make sure the virtual machine you are using to generate CP has sufficient virtual storage.
- Specify the amount of storage you want reserved for a virtual=real area.

"Part 2: Defining Your VM/370 System" describes the Directory program, including information about the VIRT=REAL operand of the OPTION control statement.

RESERVING DASD SPACE FOR A CP NUCLEUS WITH A VIRTUAL=REAL AREA

A CP nucleus with the virtual=real option requires more DASD space for system residence than a CP nucleus without the option. Use the

following formulas to calculate the number of cylinders needed for system residence (disregard any remainders):

```
Number of 2314/2319 cylinders = (96+(VRSIZE/4))/32
Number of 3330/3333 cylinders = (114+(VRSIZE/4))/57
Number of 2305/3340 cylinders = (96+(VRSIZE/4))/24
```

where VRSIZE is the size of the virtual=real storage area (in K bytes). K represents 1024 bytes. This size must be at least 32K bytes.

The number of cylinders you calculate here is the number you reserve on your system residence device. You need this information to code the SYSRES macro of your CP System Control file correctly.

If you do not reserve enough DASD space for your CP nucleus, when it is written on the system residence volume your nucleus overlays other cylinders.

VIRTUAL STORAGE REQUIREMENTS

When you are generating VM/370 you have two constraints on the maximum virtual=real size you can specify: real storage and virtual storage. The 2314 or 3330 starter system virtual machine has a maximum storage size of 1M, which means no matter how large your real machine is the largest virtual=real area you can specify is 512K. If you want a virtual=real area larger than 512K you must log on a virtual machine that has sufficient virtual storage before you load the CP nucleus. This process is described in Step 17 of the 2314, 3330 and 3340 system generation procedures.

Before you load the CP nucleus, you must be sure the virtual machine you are using has enough virtual storage to contain:

- CMS (320K)
- CP nucleus size (including the virtual=real area)

If your virtual machine does not have enough virtual storage, redefine storage and IPL again before continuing.

SPECIFYING THE AMOUNT OF VIRTUAL=REAL SPACE

If you are generating your VM/370 system to include a virtual=real machine, in Step 16 of the system generation procedure you respond "yes" to the system message:

```
VIRTUAL=REAL OPTION REQUIRED (YES,NO):
```

You are then prompted to enter the size of the virtual=real machine size:

```
STORAGE SIZE OF VIRT=REAL (MINIMUM IS 32K):
```

Normally, you would not want to specify the largest virtual=real machine possible, since that would leave few page frames available for other virtual machines.

At IPL time, the virtual=real area is locked in storage immediately following CP page 0. The system operator can issue the UNLOCK command with the VIRT=REAL option to free the virtual=real area for additional dynamic paging space for other virtual machines. The area cannot be relocked; it remains unlocked until another system IPL.

The following table shows the maximum size virtual=real area you can specify for some real machine sizes.

Real Machine Size	Maximum VIRT=REAL Size
384K	136K
512K	264K
768K	516K
1M	768K
2M	1768K

384K is the minimum real machine size on which you can have a virtual=real machine.

Calculate the maximum amount of virtual=real storage available on your real CPU as follows:

- Use Formula 1 to calculate the amount of real storage available after CP required storage is allocated.
- Use Formula 2 to calculate the maximum virtual=real size for any real machine size.

Calculating Available Real Storage (Formula 1)

Calculate the amount of real storage available (ARS) by subtracting the amount of storage required by CP from the real machine size. Formula 1 is:

$$ARS = RM - \left\lceil I + T + 12K + 4K \right\rceil_{\left\lceil \frac{RM - 256K}{64K} \right\rceil}$$

where:

RM is the real machine size.

I is the amount of storage needed to IPL CP. Refer to the load map produced when the CP nucleus is generated. The amount of storage needed to IPL CP is all of storage up to, and including, the module DMKSAV.

T is the amount of storage allocated for the CP internal trace table. CP allocates 4K of storage for each 256K of real storage for the CP internal trace table:

$$4K \left\lceil \frac{RM}{256K} \right\rceil$$

If the calculation RM/256K results in a fraction, the result is rounded upward to the next higher integer.

12K is the amount of fixed free storage allocated for the first 256K of real storage.

$$4K\left\lceil \frac{RM - 256K}{64K} \right\rceil$$

is the amount of fixed free storage allocated for real storage beyond the first 256K.

The result obtained from Formula 1 is the amount of available real storage (ARS) for a particular real machine size. This result is needed to calculate the maximum size of a virtual=real area in Formula 2.

Calculating the Maximum Size of the Virtual=Real Area (Formula 2)

Calculate the maximum size of the virtual=real area for a particular real machine size by recalculating the amount of real storage required by CP and subtracting that value from the real machine size. When you calculate the amount of real storage required by CP this time, you do not permanently allocate free storage for the portion of storage that is available for the virtual=real area (according to Formula 1). The result of Formula 2 is the maximum size virtual=real area (VRS) you can specify for a particular real machine size. Formula 2 is:

$$VRS = RM - \left\lceil I + T + 12K + 4K \right\rceil \left\lceil \frac{RM - 256K - ARS}{64K} \right\rceil$$

Use the same value for RM, I, and T as you used in Formula 1. ARS (the available real storage) is the result calculated from Formula 1. If the calculation

$$\frac{RM - 256K - ARS}{64K}$$

results in a negative value, a virtual=real area is not supported (see Example 2). If the same calculation results in a fraction, disregard the fraction (that is, round the result down to the next lower integer; see Example 3).

Example 1

Determine the maximum size of the virtual=real area for a real machine with 768K of storage that executes a VM/370 system that requires 228K to IPL.

Formula 1

$$ARS = 768K - \left\lceil 228K + 4K \right\rceil \left\lceil \frac{768K}{256K} \right\rceil + 12K + 4K \left\lceil \frac{768K - 256K}{64K} \right\rceil$$

```
ARS = 768K - [228K + 12K + 12K + 32K]

ARS = 768K - 284K

ARS = 484K
```

Formula 2

```
                 ┌                      ┌                    ┐┐
                 │                      │768K - 256K - 484K││
VRS = 768K -     │228K + 12K + 12K + 4K│------------------││
                 │                      │        64K       ││
                 └                      └                    ┘┘

                 ┌          ┌   ┐┐
                 │          │28K││
VRS = 768K -     │252K + 4K│---││
                 │          │64K││
                 └          └   ┘┘

VRS = 768K - [252K + 4K[0]]

VRS = 516K
```

Note that the fraction (28/64) resulting from the

```
RM - 256K - ARS
---------------
      64K
```

calculation in Formula 2 is rounded down to the next lower integer, zero.

Example 2

Determine the maximum size of the virtual=real area for a real machine with 320K of real storage. The VM/370 system requires 228K to IPL.

Formula 1

```
                 ┌          ┌    ┐            ┌           ┐┐
                 │          │320K│            │320K - 256K││
ARS = 320K -     │228K + 4K│----│ + 12K + 4K│-----------││
                 │          │256K│            │    64K    ││
                 └          └    ┘            └           ┘┘

ARS = 320K - [228K + 4K[2] + 12K + 4K]

ARS = 320K - [252K]

ARS = 68K
```

Note that the fraction 320/256 in the trace table calculation is rounded to the next higher integer, two.

Formula 2

```
                 ┌                      ┌                  ┐┐
                 │                      │320K - 256K - 68K││
VRS = 320K -     │228K + 8K + 12K + 4K│-----------------││
                 │                      │       64K       ││
                 └                      └                  ┘┘
```

$$VRS = 320K - \left\lceil 248K + 4K \left\lceil \frac{-4}{64} \right\rceil \right\rceil$$

Because the calculation

$$\frac{RM - 256K - ARS}{64K}$$

results in a negative value (-4/64), a virtual=real area is not allowed for this size real machine. In fact, 384K is the minimum real machine size that supports a virtual=real area.

Example 3

Determine the maximum size virtual=real area for a real machine with 1792K of real storage. The VM/370 system requires 228K to IPL.

Formula 1

$$ARS = 1792K - \left\lceil 228K + 4K \left\lceil \frac{1792K}{256K} \right\rceil + 12K + 4K \left\lceil \frac{1792K - 256K}{64K} \right\rceil \right\rceil$$

$$ARS = 1792K - [228K + 4K[7] + 12K + 4K[24]]$$

$$ARS = 1792K - [228K + 28K + 12K + 96K]$$

$$ARS = 1792K - [364K]$$

$$ARS = 1428K$$

Formula 2

$$VRS = 1792K - \left\lceil 228K + 28K + 12K + 4K \left\lceil \frac{1792K - 256K - 1428K}{64K} \right\rceil \right\rceil$$

$$VRS = 1792K - \left\lceil 268K + 4K \left\lceil \frac{108}{64} \right\rceil \right\rceil$$

$$VRS = 1792K - [268K + 4K]]$$

$$VRS = 1520K$$

Note that the fraction (108/64) resulting from the

$$\frac{RM - 256K - ARS}{64K}$$

calculation in Formula 2 is rounded down to the next lowest integer, one.

VIRTUAL MACHINE ASSIST FEATURE

The virtual machine assist feature is a combination of a CPU feature and VM/370 programming. It improves the performance of VM/370. Virtual storage operating systems which run in problem state under the control of VM/370 use many privileged instructions and SVCs that cause interrupts which VM/370 must handle. When the virtual machine assist feature is used, many of these interrupts are intercepted and handled by the CPU; and, consequently, VM/370 performance is improved.

The virtual machine assist feature is available as a hardware feature on the System/370 Models 135, 145, and 158 and as an RPQ on the Model 168.

Certain interrupts must be handled by VM/370. Consequently, the assist feature is not available under certain circumstances. VM/370 turns off the assist feature in a virtual machine if it:

- Has shared segments
- Has an instruction address stop set
- Traces SVC and program interrupts

If you IPL a virtual machine operating system that has shared segments, the assist feature is turned off. If you later IPL an operating system that can run with the assist feature, CP turns the assist feature on again.

Since an address stop is recognized by an SVC interrupt, VM/370 must handle SVC interrupts while address stops are set. Whenever you issue the ADSTOP command, VM/370 turns off the SVC handling portion of the assist feature for your virtual machine. The assist feature is turned on again after the instruction is encountered and the address stop removed.

Whenever a virtual machine issues a TRACE command with the SVC, PRIV, BRANCH, INSTRUCT, or ALL operands, the virtual machine assist feature is turned off for that virtual machine. The assist feature is turned on again when the tracing is completed.

If the virtual machine assist feature is available on a CPU, the operator can turn the feature off, and on again, for the entire VM/370 system. Also, if the feature is available to VM/370, each virtual machine operator can turn the feature off, and on again, for his own virtual machine. When you create your VM/370 directory, you can set off the SVC-handling portion of the virtual machine assist feature for various virtual machines by specifying SVCOFF on the OPTION control statement.

CMS Planning Considerations

The Conversational Monitor System (CMS) is a component of VM/370 that provides a comprehensive set of conversational facilities to virtual machine users. CMS operates only in a virtual machine, and together with CP provides a time-sharing system suitable for program development, problem solving, and general time-sharing work.

CMS is a required component of VM/370. You must generate CMS in order to support CP.

This section contains the following information about CMS:

- Storage Requirements
- Device Support
- Libraries
- Command Language
- Program Language Facilities
- Limited Support of DOS and OS
- File Management
- Disk Management
- Tape Support
- Unit Record Support
- Editing
- Batch Facility
- Saving CMS

CMS STORAGE REQUIREMENTS

CMS requires virtual storage and auxiliary storage.

VIRTUAL STORAGE

A minimum of 320K bytes for a CMS virtual machine, distributed as follows:

- CMS nucleus -- 128K
- Loader tables -- 8K
- CMS user area -- 184K (for application programs or CMS disk-resident commands)

AUXILIARY STORAGE

The CMS auxiliary storage requirements are:

- System residence for CMS -- 110 cylinders on a 2314 or 2319, 76 cylinders on a 3330 or 3333, or 203 cylinders on a 3340.

- Resident disk space for application programs (CMS commands, user programs, IBM Program Products) -- the amount of space needed is program dependent, and must be assigned by you.
- Work space for application programs (CMS commands, user programs, IBM Program Products) -- the amount of space is program dependent, and must be assigned by you.

DEVICE SUPPORT

CMS supports the virtual machine devices shown in Figure 2.

Virtual IBM Device	Virtual Address[1]	Symbolic Name	Device Type
3210,3215,1052	cuu[2]	CON1	System console
2314,2319,3330,3340	190	DSK0	System disk (read-only)
2314,2319,3330,3340	191[3]	DSK1	Primary disk (user files)
2314,2319,3330,3340	cuu	DSK2	Disk (user files)
2314,2319,3330,3340	cuu	DSK3	Disk (user files)
2314,2319,3330,3340	192[3]	DSK4	Disk (user files)
2314,2319,3330,3340	cuu	DSK5	Disk (user files)
2314,2319,3330,3340	cuu	DSK6	Disk (user files)
2314,2319,3330,3340	cuu	DSK7	Disk (user files)
2314,2319,3330,3340	19E[3]	DSK8	Disk (user files)
2314,2319,3330,3340	cuu	DSK9	Disk (user files)
1403,3211,1443	00E	PRN1	Line printer
2540,2501,3505	00C	RDR1	Card reader
2540,3525	00D	PCH1	Card punch
2401,2402,2403,2415, 2420,3410,3411,3420	181-4	TAP1-TAP4	Tape drives

[1]The device addresses shown are those that are pre-assembled into the CMS resident device table. You can change these addresses using the CP DEFINE command.

[2]The virtual address of the system console may be any valid multiplexer address.

[3]The virtual device address (cuu) of a disk for user files can be any valid System/370 device address, and can be specified by the CMS user when he activates a disk. If the user does not activate a disk immediately after loading CMS, CMS automatically activates the user's primary disk at virtual address 191, the D-disk at 192, and the Y-disk (a read-only extension of the system disk) at 19E.

Figure 2. Devices Supported by a CMS Virtual Machine

Under CP, unit record devices and the system console are simulated and mapped to different addresses and different devices. For instance, CMS expects a 3215, 3210, or 1052 type of operator's controle, but many terminals are 2741s or 3270s. Regardless of the real device type, the virtual system console is a 3215. The control program of VM/370 (CP) handles all channel program modifications necessary for this simulation. CMS virtual disk addresses are mapped by CP to different real device addresses.

The CMS system disk, normally located at virtual address 190, is read-only and contains the CMS nucleus functions and disk-resident CMS command modules. The CMS nucleus is loaded into virtual storage when you issue the CP IPL command. CMS remains resident until another IPL command is entered or until you log off. The disk-resident modules are loaded into virtual storage only when their services are needed.

The A-disk is a read/write disk and is the primary or first disk. Files that you wish to retain for later use are stored on one of your disks. Information stored on your disk remains there until you erase it. An exception is the temporary disk; files written on this disk are lost when you log off. In addition to the system disk (S-disk) and primary disk (A-disk), each CMS user can have up to eight additional disks.

You can enter CMS commands and input files from the terminal and direct output files, program results, and error and prompting messages back to the terminal.

The virtual card reader is used as the input medium for files, source decks, and data to be processed by your programs. The virtual card punch is used for your output files, language processor object decks, and various other types of data. The virtual printer is used for program results, storage dumps and language processor output.

Under VM/370, the unit record equipment is normally spooled. CMS supports only spooled unit record devices.

The following is an example of a VM/370 directory entry for a CMS virtual machine.

```
USER USER1 PASSWORD
 ACCOUNT NUMBER BIN7
  IPL CMS
  CONSOLE 009 3215
  SPOOL C 2540 READER A
  SPOOL D 2540 PUNCH A
  SPOOL E 1403 A
  LINK CMSSYS 190 190 R
  MDISK 191 2314 71 10 UDISK1 W RPASS WPASS
```

This entry describes the configuration when you log on as USER1. The Directory program control statements are described in Part 2. Briefly, this entry describes the USER1 virtual machine: it has a console at 009, a class A reader at 00C, a class A punch at 00D, a class A printer at 00E, a link to the CMS system disk (owned by userid CMSSYS) at 190, and a minidisk at 191. Once you are logged on you can change the configuration and also the spooling classes of the unit record devices. You can add devices to the VM/370 directory or dynamically add to the configuration as needed.

CMS LIBRARIES

CMS updates simulated partitioned data sets which contain:

- CMS and OS macros to be used at assembly time (macro libraries)
- Object routines to be referred to at execution-load time (text libraries)

The system macro libraries, located on the CMS system disk, are:

Library	Contents
CMSLIB MACLIB	All of the CMS macros.
OSMACRO MACLIB	The selected OS macros from SYS1.MACLIB that are supported under CMS.
OSMACRO1 MACLIB	The remaining distributed OS macros from SYS1.MACLIB.
TSOMAC MACLIB	The OS macros distributed in SYS1.TSOMAC.

If you are a DOS user in a DOS virtual machine, you can punch the DOS macros on the virtual punch and create a new CMS macro library called DOSMACRO MACLIB. This library may be used to assemble DOS programs directly under CMS. However, DOS programs do not execute under CMS.

The system text libraries, also located on the CMS system disk, are:

Library	Contents
CMSLIB TXTLIB	The CMS system text library.
TSOLIB TXTLIB	Selected TSO routines necessary to support certain features of the language Program Products.

Execution-time libraries are available with the Program Product language processors and execute under CMS.

You can generate your own libraries and add, delete, or list entries in them via the MACLIB and TXTLIB commands. You can also specify which libraries (system and user) to use for program compilation and execution via the GLOBAL command, which allows up to eight libraries to be specified.

CMS COMMAND LANGUAGE

The CMS command language lets you converse with CMS. With this command language, you can use:

- Language compilers
- An Assembler
- CMS file management system
- Context editing and line editing
- Execution control
- Debugging capability

Additionally, you can invoke the CP commands available to all virtual machines under VM/370 directly from CMS. Using these CP commands, you can send messages to the operator or to other users, dynamically change your virtual machine's configuration, and invoke spooling facilities. In CMS, the facilities of CP and CMS together appear as those of a single integrated system.

To use CMS, you must first gain access to a virtual machine via the CP LOGON command, and IPL CMS. Then you can enter commands or data from the remote terminal (virtual operator's console). Each command, upon completion, returns control to you. For information about how to use CMS and for a description of all CMS commands, see the VM/370: Command Language Guide for General Users.

CMS PROGRAM LANGUAGE FACILITIES

The languages available under CMS include:

- S/370 Assembler
- VS BASIC
- PL/I
- FORTRAN IV
- Full American National Standard COBOL

The Assembler is distributed with VM/370. The language compilers that are program products must be ordered separately. For a complete list of language processors that can be executed under CMS, see "Appendix A. IBM Programs Executable Under CMS."

CMS executes the compilers via interface modules. CMS commands are provided to invoke the compilers within the conversational environment of CMS.

COBOL programs, using the following facilities, can be compiled under CMS, but must be transferred to a machine (virtual or real) running OS for execution.

- QSAM & BDAM spanned records
- ISAM
- RERUN statement
- label handling options
- OPEN REVERSED
- Sort feature
- Segmentation feature
- ASCII code feature
- Forced end of volume
- TCAM feature

PL/I programs, using the following facilities, can be compiled under CMS, but must be transferred to a machine (virtual or real) running OS for execution.

- Multi-tasking
- Teleprocessing file support
- ISAM
- Backwards attribute
- Spanned records for buffered files
- Sort-merge
- Checkpoint-restart
- ASCII data sets
- Track overflow

CMS TEXT PROCESSING FACILITY

Text processing facilities that can create formatted output from one or more CMS files containing text and/or control words are available through the SCRIPT command. SCRIPT/370 is an IBM Installed User Program that must be ordered separately.

LIMITED SUPPORT OF OS AND DOS IN CMS

Object programs (TEXT files) produced under CMS and under OS in real or in virtual machines may be executed under CMS if they do not utilize certain OS functions not simulated by CMS. Object programs using non-simulated OS macro functions must be transferred to an appropriate real or virtual OS machine for execution.

Sequential and partitioned data sets residing on OS disks can be read by OS programs running under CMS. Also, certain CMS commands can be used to process data sets on OS disks.

CMS commands, and programs that use OS macros simulated by CMS, can read sequential DOS files. No DOS macros are simulated; DOS files are treated as if they were OS data sets and are read by OS macros.

The application programmer who normally uses CMS to interactively create, modify, and test his programs may require facilities not supported in CMS (for example, an OS program using ISAM). He can alternately execute CMS and another operating system in the same virtual machine.

A description of the actual processes for reading OS or DOS files and of alternating operating systems is in the VM/370: Command Language Guide for General Users.

CMS FILE MANAGEMENT

CMS provides interfaces with virtual disks, tapes, and unit record equipment. Most CMS commands assume that files are stored on disk. This means that files stored on media other than disk must be transferred to disk before you can issue a CMS command that uses them. Similarly, most CMS commands that create files do so on disk (such as listing files from a compilation). In many cases you may want to transfer these files to cards or tape, or to print them.

CMS DISK MANAGEMENT

CMS can manage up to ten virtual disks (created on 2314, 2319, 3330, or 3340 storage devices) for each user. One of these is the CMS system disk (S-disk), a read-only minidisk, which normally has a device address of 190. Your files may be accessed from up to nine active CMS disks. Usually, one of these is a read/write disk called the A-disk with a device address of 191.

Each CMS disk has a six-character label (or volid) associated with it. Record 3 (on cylinder 0, track 0), of a CMS disk includes a ten-byte label, consisting of the following:

- Four characters: CMS=
- Six characters: desired label
  (blank filled if less than 6 characters; truncated if more than 6 characters)
- The remaining bytes of the record are all binary zeros.

The CMS FORMAT command is used to initialize a disk area in the CMS format and to write a label onto a disk. All CMS disks must be formatted before being used for the first time.

Disks do not need to be formatted for each session unless the disk is a temporary disk dynamically added to the virtual machine configuration via the CP DEFINE command. Dynamically defined disks must be formatted each time they are added to the virtual machine configuration.

Each virtual disk has an automatically updated CMS file directory (called the Master File Directory) that keeps an inventory of all files on that disk. The file directory is updated after each CMS command that changes the status of files on that disk. A file directory has a name in the form of a single alphabetic letter A, B, C, D, E, F, G, S, Y, or Z, which corresponds to the mode letter of the ten possible user disks.

The relationship between a file directory on a physical disk and the file directory name is established when you issue the ACCESS command.

The data extent of a particular CMS file is limited to one virtual disk (up to 12,848,000 bytes). The file management system limits the number of files on any one virtual disk to 3400 (for 3330/3333/3340 devices) or 3500 (for 2314/2319 devices), and the number of records in a particular CMS file to 65,535.

FILE MANAGEMENT

CMS disks are formatted into 800-byte physical records or blocks. A 3330 disk volume holds 14 physical blocks per track, a 3340 disk volume holds 8 physical blocks per track, and a 2314 or 2319 disk volume holds 15 physical blocks for every two tracks.

Logical records are imposed upon constant physical blocks as shown in Figure 3. Logical records can span physical blocks.

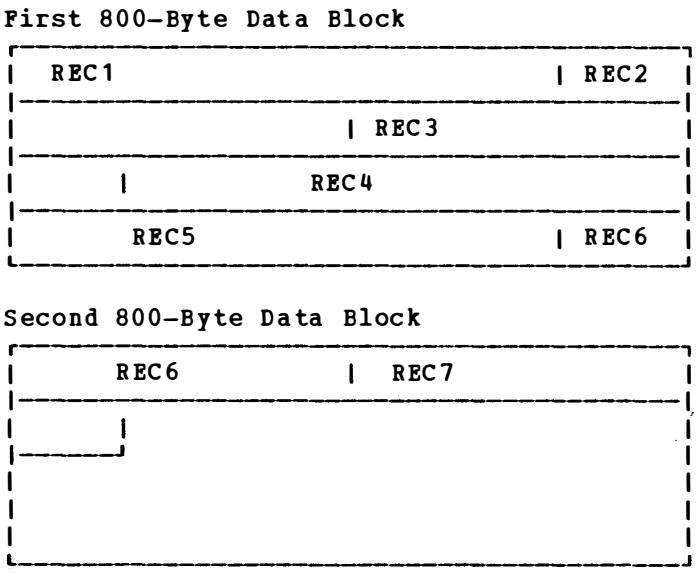


Figure 3. File Format of Logical Records

Data records may be fixed-length or variable-length, but a mixture within a file is not permitted. Where necessary, items are begun in one block and continued in the next, for example, record 6 in Figure 3. Physical blocks are randomly assigned to a file as space is required, and released when no longer in use. The physical blocks assigned to a file can be discontiguous, and are chained together by a series of pointers called a chain link, anchored in a File Status Table. Note that the entire 800 bytes of a data block are used.

Space for required files (compiler work files, LISTING files, and object modules), as well as user created files, is automatically allocated by CMS. As a file grows, its space is expanded, and it is contracted if the space requirements are reduced.

DISK ACCESS

Disks can be accessed in two ways: read-only, where files on that disk can only be read; and read/write, where files can be read and written.

To access a disk, you must:

- Identify a disk as part of your virtual machine configuration. This disk is available if it is defined in your VM/370 directory entry, or it can be acquired dynamically by issuing the CP LINK or DEFINE command.
- Identify the disk to CMS and assign it a file directory name. Use the ACCESS command after you load CMS. The CMS ACCESS command associates a particular disk with a given file directory name and, optionally, specifies which files on the disk are to be used and specifies the disk as read-only.

Both CP and CMS can control read/write access to disks, as is illustrated in Figure 4.

	CMS ACCESS	
CP ACCESS	Read-only	Read/write
Read-only	Read-only	Read-only
Read/Write	Read-only	Read/Write

Figure 4. CP and CMS Disk Access

The access allowed by CP is determined by your VM/370 directory entry or the form of the LINK command you issue for a particular disk.

The disk access allowed by CMS is determined by the CMS ACCESS command.

FILE SHARING IN CMS

Minidisks can be shared among CMS users via the VM/370 disk sharing facilities. Since CMS does not provide any control for multiple writes (ENQ, DEQ), this form of CP sharing is not recommended.

Each CMS file has an access mode associated with it at file creation. The filemode number of the file identifier determines the access mode. The following modes apply to CMS files:

- Private
- Read/write
- Read/erase

If you have read-only access to a disk, you cannot read a private file but can read all other files. However, if you have read/write access to a disk, you can read and write all the files, including those designated as private.

Read/erase is a filemode normally used by the language processors for temporary work files. Once the file is read, it is automatically erased.

CMS TAPE SUPPORT

Each CMS machine can support up to four magnetic tape units at virtual addresses 181, 182, 183, and 184. They may be either 2401, 2402, 2403, 2415, 2420, 3410/3411, or 3420 drives, or a mixture of tape drives.

Each tape-handling command allows you to specify the mode of the tape (7-track or 9-track), density, and tape recording technique: odd parity, converter on, translator off, or odd parity, converter off, translator on.

If you do not specify the mode for a 7-track tape, CMS issues a mode set indicating 7-track, 800 bpi (bits per inch), odd parity, converter on, and translate off. If the tape is 9-track, the mode is ignored and is assumed to be 1600 bpi for dual density drives and 800 bpi for single density drives.

As an alternative to specifying mode in each tape command, you can issue a CMS TAPE command that sets the mode for the tape and stays in effect until reissued. You must do this if one of your programs is to use tapes in other than the default mode.

With one exception, CMS commands permit only unlabeled tapes to be read or written. However, the CMS TAPPDS command can read standard OS tape labels. Your programs executing under CMS must use unlabeled tapes or provide code to create and read their own labels as data records.

Multivolume tape files are not supported by CMS.

Note: These restrictions only apply when you run CMS. DOS and OS systems running in virtual machines can continue to read and write tapes with standard labels, non-standard labels, and no labels on single and multireel tape files.

The VM/370 operator must attached tapes to your CMS virtual machine before any tape operation can take place.

CMS UNIT RECORD SUPPORT

CMS supports one virtual card reader at virtual address 00C, one virtual card punch at virtual address 00D, and one virtual printer at virtual address 00E. Under VM/370, these devices are spooled. CMS does not support real or dedicated unit record devices, nor does it support a virtual 2520 Card Punch. Figure 2 in this section lists the devices supported as virtual devices by CMS.

CARD READER

The READCARD command reads data records from the spooled card reader to a CMS disk. Input records of 151 or fewer characters are accepted. Column binary data is not acceptable. Your card decks must be read into the virtual reader before a READCARD command can be issued. You can do this in one of two ways:

- Place a card deck, containing only one file, in a real card reader and have CP read it. The card images are placed on a spool file in the specified virtual machine's virtual card reader. If you are not logged on when data is spooled to your virtual card reader, the deck remains in your virtual card reader until you log on and issue the READCARD command.
- Transfer records from your virtual card punch or printer to a virtual card reader (your own or that of another virtual machine).

For more information on reading files from the CMS card reader, see the VM/370: Command Language Guide for General Users.

CARD PUNCH

The CMS PUNCH command causes the specified file to be punched to the spooled card punch. Records up to 80 characters long are accepted. Shorter records are padded to 80 characters with blanks filled in to the right. The following are not supported:

- Punch stacker select
- Punch feed read
- 3525 Multiline Card Print Feature

PRINTER

The PRINT command prints the specified disk file on the spooled printer. You can specify whether the first character of each record is to be interpreted as a carriage control character or as data. Both ASA and machine code carriage control characters are supported. The file may have either fixed or variable length records.

EDITING

Using the CMS Editor, you can create new files online or modify or display portions of existing files.

The CMS Editor operates on fixed and variable length records. The maximum record length accepted by the editor is 160 characters. You can specify file characteristics, such as record length, format, and tab locations. The system includes defaults for certain filetypes (such as ASSEMBLE and EXEC).

Files are edited in virtual storage. The maximum size file that can be edited is determined by the amount of virtual storage available to you, minus that used by the CMS nucleus.

If the file does not fit into virtual storage, it must be divided into smaller files, and each file edited separately. Sufficient disk space must be available to accommodate both parts of the file. Alternately, you can temporarily increase the size of your virtual storage by issuing the CP DEFINE STORAGE command.

For more information about the editing facilities of CMS, see the VM/370: EDIT Guide.

CMS BATCH FACILITY

The Batch Facility is a VM/370 programming facility that executes under CMS. It allows you to execute jobs in batch mode. You can submit jobs from a virtual machine or from a real card reader. You do not have to be logged on to submit jobs to the batch virtual machine. Information about the CMS batch facility is in the VM/370: Command Language Guide for General Users and operating information is in the VM/370: Operator's Guide.

If you want to use the batch facility you must define a CMS batch virtual machine when you create your VM/370 directory. You should include a read/write disk at virtual address 195 in the directory entry. Appendix C contains a batch virtual machine entry in its representative VM/370 directory.

Also, you must have an entry in the VM/370 directory for each user that submits a job to CMS Batch. This entry can be the minimum: userid, password, and one device. For all users that submit jobs when they are not logged on, you code the password as NOLOG and do not need any devices defined in the VM/370 directory. Because users with a NOLOG password do not have devices defined for their virtual machines, CMS Batch uses the TO and FOR operands of the CP SPOOL command to transfer their files.

The Batch Facility provides two entry points so you can add support to Batch that your installation may require. BATEXIT1 is provided so you can write your own routine to check non-CMS control statements. BATEXIT2 is provided so you can process additional information on the /JOB card. These entry points are described in the VM/370: System Programmer's Guide.

You can write EXEC procedures to control the operation of a Batch Facility virtual machine. See the VM/370: EXEC User's Guide for examples of these EXEC procedures.

SAVING CMS

CMS is designed so that it can be saved easily and so that the second segment of CMS can be shared by CMS users. The VM/370 starter system has entries in the system name table (DMKSNT) and CP system control file (DMKSYS) so that you can save CMS at the end of the system generation procedure for CMS.

For more information about saved systems, see the "Saved Systems" section of this manual and see the VM/370: System Programmer's Guide.

Planning Considerations for Virtual Machine Operating Systems (Other Than CMS)

This section contains information about the following:

- Virtual Machine Options
- Multiprogramming Systems
- Multiple Virtual Machines Using the Same Operating System
- Multiple Access Virtual Machines
- The RSCS Virtual Machine
- The ASP Virtual Machine
- VM/370 Using Channel Switching
- Operating Systems Using Reserve/Release

CMS or any combination of operating systems such as OS/PCP, MFT, MVT, VS1, VS2, DOS, DOS/VS, and RSCS may be executed as operating systems in a virtual machine created and controlled by VM/370. Programs or systems to be run in a virtual machine, however, cannot include any hardware or software timing dependencies. For example, if I/O devices (or the programs that control them) require program responses within a defined time for correct data transmission, they may not work correctly. Execution of programs under VM/370 that require dynamic modification of channel programs is allowed only in certain cases: they can be executed in a virtual machine with the VM/370 performance option called virtual=real, an OS ISAM program can execute in a virtual machine if the ISAM option is specified in the VM/370 directory entry for that virtual machine, or an OS/VS TCAM program can execute in a virtual machine that has OS/VS TCAM Level 5, generated or invoked with the VM/370 option. Each operating system under VM/370 control requires that the virtual machine have an I/O configuration compatible with the one for which the operating system was originally generated.

The environment of multiple virtual machines permits several of these systems to be executing concurrently. Production work may be run in one or more virtual machines while other virtual machines are executing:

- A terminal-oriented conversational system.
- A remote spooling subsystem, which transmits bulk data to and from remote locations.
- A back-release system running application programs not upgraded to the current level of the production system.
- A current-release system with new Program Temporary Fixes (PTFs) applied.
- A new release or option being generated and tested.

The interactive execution of conventional operating systems in a virtual machine environment provides several convenient operating facilities. Some of these are:

System Programming

- Testing of new or modified Supervisor Call (SVC) routines on a virtual machine.

- Applying and testing PTFs on a virtual machine.
- Generating and testing new releases of an operating sytem in a virtual machine.
- Hands-on console debugging as though on a dedicated real machine, via a terminal device.

Application Programming

- Debugging while under operating system control, from a terminal.
- Designing of application programs without real storage constraints.
- Defining minidisks and other virtual devices to design and test a slightly different or larger machine configuration before the actual hardware is installed.

Operations

- Training of operators in their own virtual machine.
- Defining a virtual machine and its devices as backup to some other real machine.
- Permitting divergent installation requirements to be run concurrently on a single real machine.

VIRTUAL MACHINE OPTIONS

VM/370 provides several optional services to virtual machines, including:

- The REALTIMER option, which updates a virtual machine interval timer when that virtual machine is in a self-imposed wait state. This option is required for virtual machines running systems that wait for a timer interrupt to continue processing.
- The ISAM option, which allows the virtual machine to execute the self-modifying CCW command sequences generated by the OS ISAM modules in OS PCP, MFT, or MVT. This option is not required for the proper functioning of ISAM in DOS or OS/VS. However, if ISAM is run in the virtual=real area of an OS/VS virtual machine, the ISAM option is required. This option does not permit other types of self-modifying CCW sequences to function.
- The ECMODE option, which allows the virtual machine to use the complete set of virtual System/370 control registers and the dynamic address translation feature of the System/370. Programming simulation and hardware features are combined to allow use of all the available features in the hardware.
- The BMX (virtual block multiplexer) option, which allows an operating system that is running in a virtual machine to overlap multiple SIO (Start I/O) requests on a specified channel path (this is done in the

real block multiplexer environment). The selector channel mode is the normal (and default) channel mode for virtual machines. When the BMX option is invoked, it applies to all channels in the virtual machine except channel 0 or channels that have a channel-to-channel adapter (CTCA). The CP DEFINE command can redefine the channel mode for a virtual machine.

The REALTIMER, ISAM, and ECMODE options increase the amount of CP overhead incurred by the virtual machines using them, and should not be given to a virtual machine unless they are required. These options are specified in the OPTION statement in the VM/370 directory. If a particular option requires an option only occasionally, you may choose to define two virtual machines: one with the option specified, and one without; in this way, the additional overhead is incurred only when necessary.

The REALTIMER can be set off by a user with the CP SET TIMER command. Timers are described in the VM/370: System Programmer's Guide.

For more information about these and the other options (ACCT, VIRT=REAL, and SVCOFF) that can be specified in the OPTION statement, see the "Creating Your VM/370 Directory" section of Part 2.

OS ISAM CHANNEL PROGRAMS

Certain OS/PCP, MFT, and MVT ISAM channel programs use a self-modifying operation that is not allowed under normal VM/370 processing. With the ISAM option selected, CP can scan the specific OS/PCP, MFT, and MVT ISAM channel program to handle the self-modifying sequence properly.

Only those users with the ISAM option in their VM/370 directory entry have their CCW strings checked for self-modifying operation; thus not all users incur the additional CP overhead. This option is not needed for DOS or OS/VS ISAM (except when OS/VS ISAM is executed in a virtual=real area of an OS/VS virtual machine).

MULTIPROGRAMMING SYSTEMS UNDER VM/370

When a multiprogramming system such as OS or DOS is run on a virtual machine, its resource algorithms interact with those of VM/370; especially when the virtual operating system has a page wait or I/O wait. There are two methods in which multiprogramming systems interact with VM/370: one method is without the benefit of a special communication path and the second method is with the special path. The communication path, called the VM/VS Handshaking feature, is only available for OS/VS1 virtual machines.

MULTIPROGRAMMING SYSTEMS WITHOUT THE VM/VS HANDSHAKING FEATURE

Page Waits in a Virtual Machine

If, during its execution, an OS- or DOS-created task or program must wait for a CP-provided service such as a virtual storage page, the virtual machine is marked non-dispatchable even though other partitions or tasks in that virtual machine may be ready and available to run.

Those other tasks in the virtual machine cannot be dispatched by the operating system until the CP page wait is satisfied. The net effect is that the highest priority program of the virtual machine gets almost all the available CPU time, and the other programs experience significant degradation.

When multiprogramming systems must be run in a virtual machine, consider using the virtual=real option, reserved page frames, or locked pages. When the virtual=real option is used, paging is eliminated. The reserved page frames option tends to keep the most active pages in storage and the locked pages option locks the specified pages in storage.

I/O Waits in a Virtual Machine

On a real machine, when a task is waiting for an I/O operation to complete, the lower priority tasks are given use of the CPU. Under VM/370, the I/O operations of a particular virtual machine are overlapped with the CPU execution of that and other virtual machines. Consequently, lower priority tasks created by OS and DOS are given the CPU resource less frequently when executing in a virtual machine than when executing in a real machine.

The system operator can issue the CP command SET FAVORED for a specific virtual machine to ensure that its lower priority DOS or OS tasks have a chance to execute. The favored execution option reduces the effect of a variable system load on the favored virtual machine.

In addition, the assured percentage can be specified to guarantee that a certain amount of CPU time is made available to the virtual machine, provided that it is able to use the time.

The system operator can also set the priority of a virtual machine by issuing the CP command SET PRIORITY. A low value gives the virtual machine a higher priority, and ensures that it is dispatched for execution more frequently than other virtual machines.

Spooling Considerations

Most multiprogramming operating systems provide their own spooling facilities. Since VM/370 also provides these facilities, double spooling can occur. If you have a significant amount of printing or punching to do, you may think one of the spooling facilities should be eliminated. This is not necessarily true. In fact, if the multiprogramming operating system's spooling facility blocks its output (such as OS/VS1 does), the most efficient spooling arrangement is usually to let both VM/370 and OS/VS1 spool. When a virtual machine operating system blocks its output, VM/370 has fewer SIOs to process.

See the VM/370: Command Language Guide for General Users for more information about punching and printing in a virtual machine.

If a virtual machine without the handshaking feature shares real unit record devices with other virtual machines, the CP spool files are not closed until the virtual machine operator closes them. This means the CP spool files are not sent to the real printer or punch until the virtual machine operator intervenes.

THE VM/VS HANDSHAKING FEATURE

The VM/VS Handshaking Feature is a communication path between VM/370 and OS/VS1 that makes each system control program aware of certain capabilities and requirements of the other. The VM/VS Handshaking feature consists of:

- Processing VS1 pseudo page faults
- Closing VM/370 spool files when the VS1 job output from its Direct System Output (DSO), terminator, and output writers is complete
- Providing an optional nonpaging mode for VS1 when it executes under the control of VM/370
- Providing miscellaneous enhancements for a VS1 virtual machine executing under the control of VM/370

A page fault is a program interrupt that occurs when a page that is marked "not in storage" is referred to by an instruction in an active page. The virtual machine requesting the page is placed in the wait state while the requested page is brought into real storage. However, with the handshaking feature, a multiprogramming VS1 virtual machine executing under the control of VM/370 may dispatch one task while waiting for VM/370 to honor a page request for another task. When the pseudo page fault portion of VM/VS Handshaking is active, VM/370 sends a pseudo page fault (program interrupt X'14') to VS1. When VS1 recognizes a pseudo page fault, it places only the task waiting for the page in page wait and can dispatch any of its other tasks.

When the handshaking feature is active, VS1 closes the CP spool files by invoking the CP CLOSE command when the VS1 job output is complete. Once these spool files are closed, they can be processed by VM/370 without operator intervention.

VS1 can execute in nonpaging mode. Nonpaging mode exists when (1) the handshaking feature is active, and (2) the VS1 virtual storage size equals the virtual storage size of the VM/370 virtual machine. When VS1 executes in nonpaging mode, fewer privileged instructions are executed and duplicate paging is eliminated. Note that a VS1 virtual machine may have a larger working set when it is in nonpaging mode than when it is not in nonpaging mode.

Also, there are some other enhancements for VS1 executing under the control of VM/370. With the handshaking feature, VS1 avoids some of the instructions and procedures that would be inefficient in the VM/370 environment.

When the VM/VS Handshaking feature is active, the operation of VS1 with VM/370 more closely resembles the standalone operation of VS1, because:

- One VS1 task can be dispatched while another is waiting for a page to be brought into real storage.
- There is less need for the virtual machine operator to intervene because output files are automatically closed and processed.

Also, a VS1 operating system with the VM/VS Handshaking feature can execute by itself on a real machine or under the control of VM/370. Although handshaking is a system generation feature for VS1, it is active only when VS1 executes under the control of VM/370; it is disabled when that same VS1 operating system is run on a real machine. The VM/VS Handshaking feature is not active unless:

- VS1 is generated with the VM/370 option.

- VS1 is run under the control of a version of VM/370 that supports the feature. The VM/VS Handshaking feature is available with Release 2 PLC 13 of VM/370.

Even when the handshaking feature is active for an OS/VS1 virtual machine, the pseudo page fault portion of the handshaking feature is not available until it is set on with the CP SET PAGEX ON command. The CP SET command can set pseudo page fault handling on and off.

MULTIPLE VIRTUAL MACHINES USING THE SAME OPERATING SYSTEM

In general, an operating system which is to run in a virtual machine should have as few options generated as possible. This is also true when several virtual machines share a system residence volume. Very often, options that improve performance on a real machine have no effect (or possibly an adverse effect) in a virtual machine. For example, seek separation, which improves performance on the real machine, is redundant in a virtual machine: CP itself issues a standalone seek for all disk I/O.

Other Virtual Machines

Sharing the system residence volume avoids the necessity of having to keep multiple copies of the operating system online. The shared system residence volume should be read-only.

The CMS system residence volume is already read-only so nothing special need be done to share it among virtual machines. The OS system can be shared among users is all data sets with write access are removed from the system residence volume. The DOS system residence can be shared among virtual machines if each virtual machine has a unique standard label cylinder and its own read/write private core image library where it can catalog programs. Some change to DOS is necessary to support shared system residences. Refer to the VM/370: System Programmer's Guide for more details.

MULTIPLE-ACCESS VIRTUAL MACHINES

Multiple-access programs are those which execute in a virtual machine and directly control terminals. These terminals need not be of the type supported by CP as virtual operator consoles, but may be of any type supported by the virtual machine. The lines used are either dedicated to the virtual machine via the directory entry or assigned to the virtual machine dynamically.

A subset of the lines of a real transmission control unit (TCU) can be defined as a virtual TCU for a virtual machine, as shown in Figures 5 and 6.

Figure 5 shows two multiple-access systems (controlled by virtual machines VM1 and VM2). Each controls real 3277s by using part of the resources of the real 3272, but it appears to both virtual machines as if they each have sole control of the real 3272. (The virtual system consoles of VM1 and VM2 are not shown.)

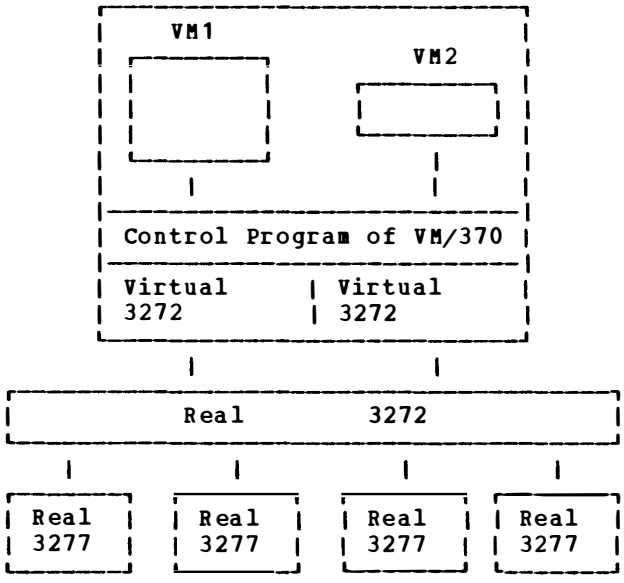


Figure 5. Virtual Devices: 3270 Terminals

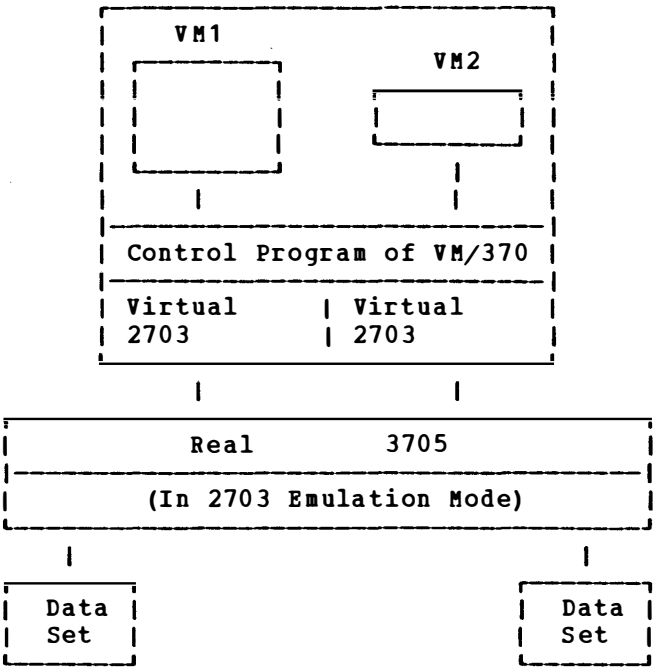


Figure 6. Virtual Devices: Teleprocessing Units

In Figure 6, two lines on the real 3705 are defined as virtual transmission control units for two virtual machines named VM1 and VM2. The remaining lines may support virtual operator consoles.

The virtual machine operating system may be one like APL DOS-360 (with the CP option invoked) that supports a number of remote terminals, as shown in Figure 7.

When the terminals supported by the virtual machine's operating system are of the same type as those supported by VM/370 as virtual system consoles, a real line can be assigned to a virtual TCU.

To do this, virtual lines are defined in the virtual machine's VM/370 directory entry via the SPECIAL record, or are added to the logged-on virtual machine via the CP DEFINE command.

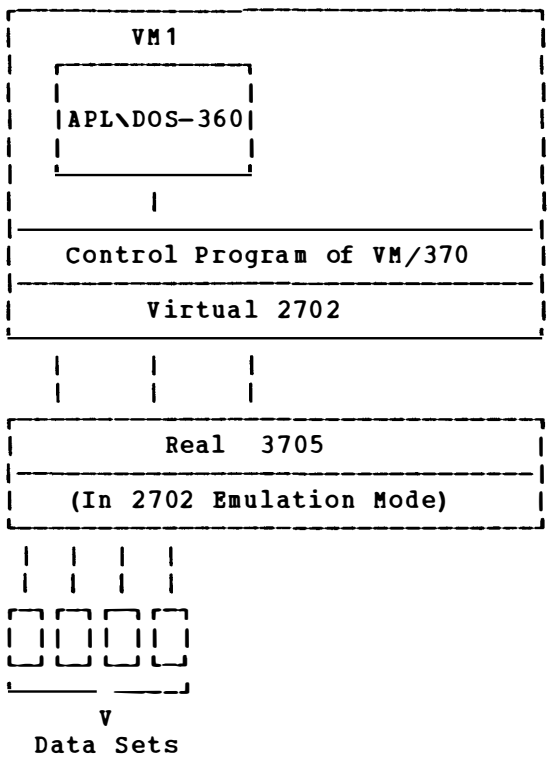


Figure 7. A Virtual Multiple-Access System

Figure 8 illustrates a VM/370 directory entry for a multiple-access virtual machine to run APL\DOS-360 (with the CP option).

```
USER   APLSYS 051243   256K
  CONSOLE 01F   3215
   SPOOL   00C   2540     READER
   SPOOL   00D   2540     PUNCH  B
   SPOOL   00E   1403     A
   DEDICATE 190 SYSRES
   DEDICATE 191 SYSWRK
   SPECIAL 080   2702     IBM
   SPECIAL 081   2702     IBM
   SPECIAL 082   2702     IBM
   SPECIAL 083   2702     IBM
```

Figure 8. VM/370 Directory Entry for an APL\DOS-360 Virtual Machine

You can use the CP DIAL command to connect a terminal supported by both CP and a multiple-access system. Such terminals can be on either nonswitched or switched lines. To connect to the virtual machine in Figure 8, you would enter the command as follows:

```
DIAL APLSYS
```

The CP system matches the terminal type with an equivalent virtual line that is available and enabled (in this example, 080, 081, 082, or 083). Once a connection is made, the terminal is controlled by the virtual machine to which it is logically connected (in this example, the APL\DOS-360 virtual machine), and neither you nor the terminal can issue any CP or CMS commands. It remains connected until you log off APL using standard APL logoff procedures, or until the APL virtual machine is logged off. You are then free to log on CP or to DIAL another multiple-access system.

Dial-up terminals supported by the multiple-access system may be of a different type than those supported by VM/370 as virtual system consoles. Such terminals must be on switched lines, and the CP DIAL command cannot be used. You must dial the multiple-access system's telephone numbers directly.

In addition, a communications system can be tested using multiple virtual machines in place of multiple real machines, as shown in Figure 9. The virtual 2701 units could each be defined as a one-line 2701; on the real machine there exists a single two-line 2701.

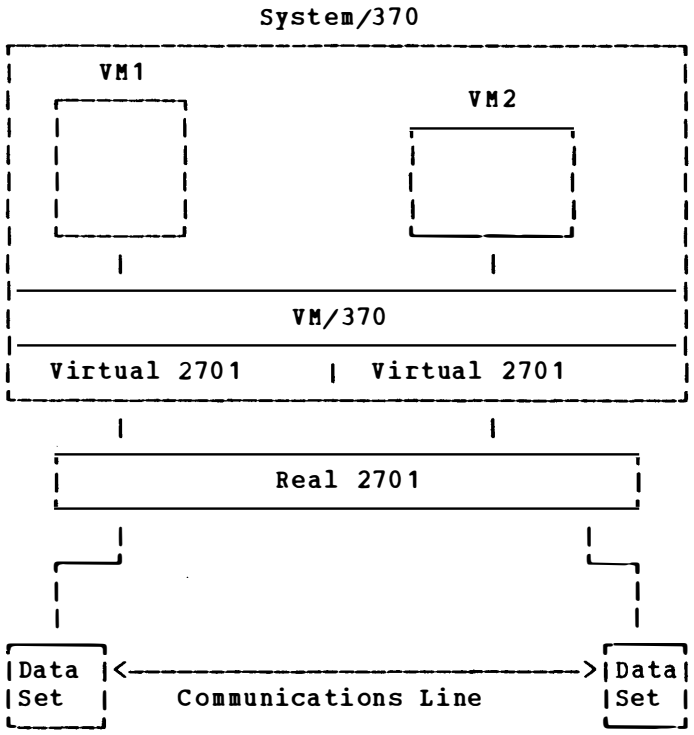


Figure 9. A Communications System Test

The communications system test in Figure 9 is based on the assumption that the real transmission control unit is equipped with the appropriate data sets and line capability.

Figure 10 illustrates a virtual transmission control unit running remote 3270-type units.

When the terminals supported by the multiple-access system are not of the type supported by VM/370 as virtual operator consoles, the real line appearances must be:

- Defined in the VM/370 directory entry for the virtual machine via the DEDICATE record, as

  ```
  DEDICATE cuu
  ```

  where cuu is the real address of the appropriate line appearance on the real transmission control unit

  — or —

- Attached to the virtual machine by an operator with privilege class B as

  ```
  ATTACH cuu TO VM1 AS vcuu
  ```

where cuu is the real address of the appropriate line appearance on the real transmission control unit, and vcuu is the address of the line appearance as generated in the virtual machine operating system.

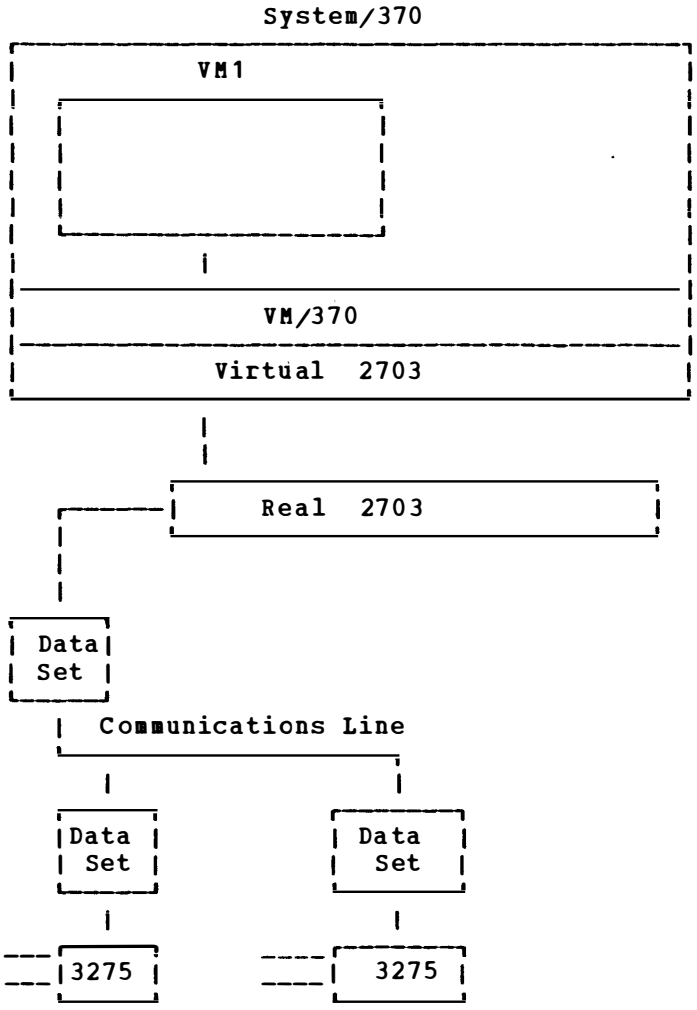


Figure 10. A Virtual TCU Running 3270-Type Units

PERFORMANCE CONSIDERATIONS

When virtual machine activity is initiated on an infrequent or irregular basis, such as from a remote terminal in a teleprocessing inquiry system, some or all of its virtual storage may be paged out before the time the virtual machine begins processing. The paging activity required so that the virtual machine can respond to the teleprocessing request may cause an increase in the time required to respond to the request.

Use the locked pages or reserved page frames options to improve performance.

If the program must be run in the dynamic paging area, then locking specific pages of the virtual machine into real storage may ease the

problem. However, it is not always easy or possible to identify which specific pages are always required.

A more flexible approach than locked pages is the reserved page frames option. When a temporarily inactive virtual machine having this option is reactivated, these page frames are immediately available. If the program code or data required to satisfy the request was in real storage at the time the virtual machine became inactive, no paging activity is required for the virtual machine to respond.

THE RSCS VIRTUAL MACHINE

The Remote Spooling Communications Subsystem (RSCS), operating in a virtual machine, handles the transmission of files between VM/370 users and remote terminals and stations. Figure 11 illustrates a typical RSCS configuration.

Three lines of the real 3705, operating in 2703 emulation mode, are shown dedicated to the RSCS virtual machine. The communication lines are shown attached to a 3780 Data Communication Terminal, a System/3, and an OS/HASP processor running in a remote System/360 or System/370.

The RSCS machine uses the spooling facilities of VM/370 as the interface between virtual users and itself. Any user who wishes to have an output file transmitted to a remote location must associate tag information (such as destination and priority) with his file via the TAG command and spool the file to the RSCS machine's virtual reader. RSCS analyzes the tag data, enables the appropriate line, and transmits the data using the line protocol required by the receiving station.

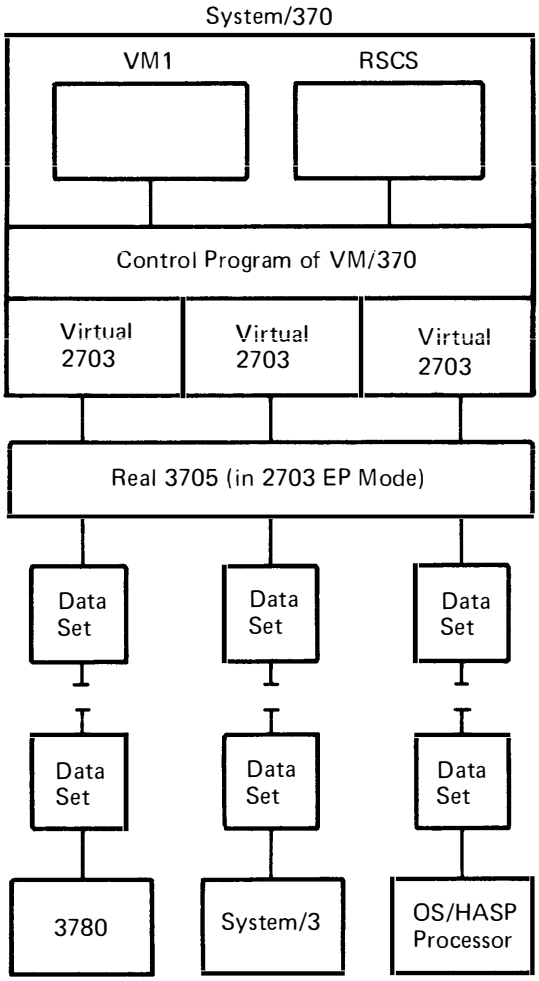
Remote locations can submit card files to the RSCS machine and address them to either a VM/370 user or to RSCS itself for transmission to another remote station. RSCS produces a VM/370 spool file by writing that data to virtual unit record devices and, if the file is destined for a VM/370 user, sends it to the user's virtual reader via the CP SPOOL command. If the file is addressed to RSCS, it is queued on RSCS's virtual reader and then handled in the same manner as a file spooled to RSCS by a VM/370 user. In this case, it is the responsibility of the remote station that originated the data to supply the tag information.

RSCS can also function as a remote workstation of a HASP/ASP type batch processor. VM/370 users and remote stations can submit jobs to RSCS for transmission to the HASP system. After processing, HASP can return printed and/or punched output to RSCS for spooling to the real system printer or punch. For more information about RSCS, see the VM/370: Remote Spooling Communications Subsystem (RSCS) User's Guide.

RSCS PLANNING CONSIDERATIONS

All the files you need to generate RSCS are on the PLC (Program Level Change) tape.

Before you perform the RSCS generation procedure, be sure you have a virtual machine defined for RSCS. The virtual machine you intend to run RSCS should have:



| Figure 11. A Remote Spooling Communications Subsystem

| • 512K of virtual storage

| • A console

| • A 5-cylinder minidisk for the RSCS system residence volume. The RSCS
| system disk must have a write password.

| You can define more than one RSCS virtual machine. Also, you can
| have more than one RSCS virtual machine running at the same time.
| However, when multiple RSCS virtual machines are running at the same
| time, each must have a unique user identification (userid) and local
| location identification (ID=linkid).

| When you code the GENLINK macros during the RSCS generation
| procedure, you must code the local location identification on the first
| GENLINK macro.

| Information about generating and installing RSCS is in "Part 3:
| Generating VM/370 (CP, CMS, and RSCS)."

THE ASP VIRTUAL MACHINE

If you use the OS Attached Support Processor (ASP) you may find virtual machines useful in two ways.

First, a virtual ASP system can be run by two virtual machines (VM1 and VM2) together with a virtual channel-to-channel adapter (CTCA). This is illustrated in Figure 12.

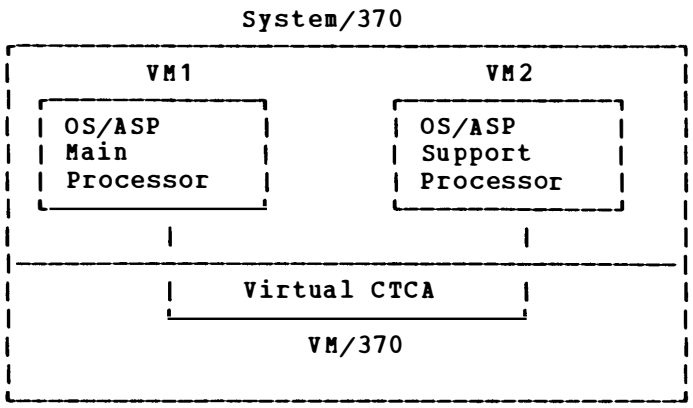


Figure 12. A Virtual Machine ASP System

The virtual ASP system (VM1 and VM2) shown in Figure 12 may be used to install and test a new ASP release, or for ASP system testing in a virtual environment concurrent with normal production. This eliminates the need to dedicate the real ASP computer for new system testing.

The VM/370 directory entry for each of the virtual ASP processors must contain a record defining a virtual channel-to-channel adapter in the form:

```
SPECIAL  280 CTCA
```

where 280 is the address of the channel-to-channel adapter as generated in the OS system.

When both virtual ASP machines are logged on VM/370, the CP COUPLE command must be issued by one of the virtual machine operators:

```
couple 280 to vm2  380
```

where 280 is the address of VM1's virtual channel-to-channel adapter, VM2 is the userid of the second virtual machine, and 380 is the address of VM2's virtual channel-to-channel adapter. After the channels are coupled, the operator of each virtual machine can then IPL his OS system and start running ASP.

Second, an additional virtual machine (VM3), with a real channel-to-channel adapter to a real System/360 or System/370, can be running as either the main or support processor in a production ASP system. This is illustrated in Figure 13.

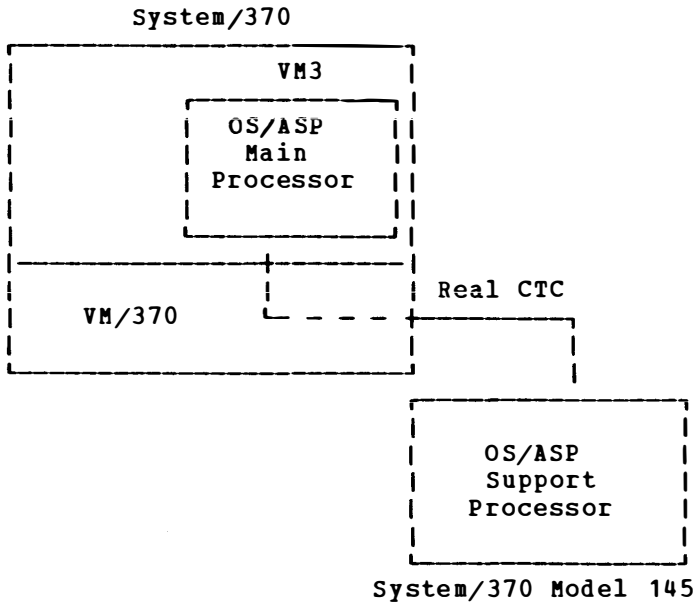


Figure 13. A Virtual Machine in an ASP Configuration

The real channel-to-channel adapter must be defined in the VM/370 system generation procedure via the RDEVICE macro with a device type of CTCA (DEVTYPE=CTCA). The virtual machine (VM3) must have this device assigned to it before the IPL. This can be done via the DEDICATE statement in the virtual machine's VM/370 directory entry as follows:

```
DEDICATE 280 380
```

where 280 is the address of the channel-to-channel adapter as generated in the OS system, and 380 is the address of the real channel-to-channel adapter as specified in the VM/370 system generation procedure.

If there is no DEDICATE statement for the channel-to-channel adapter in the virtual machine's directory entry, a resource operator with privilege class B must attach the real channel-to-channel adapter to the virtual machine.

Note: See the description of the COUPLE, DEFINE, and DETACH commands in the VM/370: Command Language Guide for General Users for further information on the virtual channel-to-channel adapter.

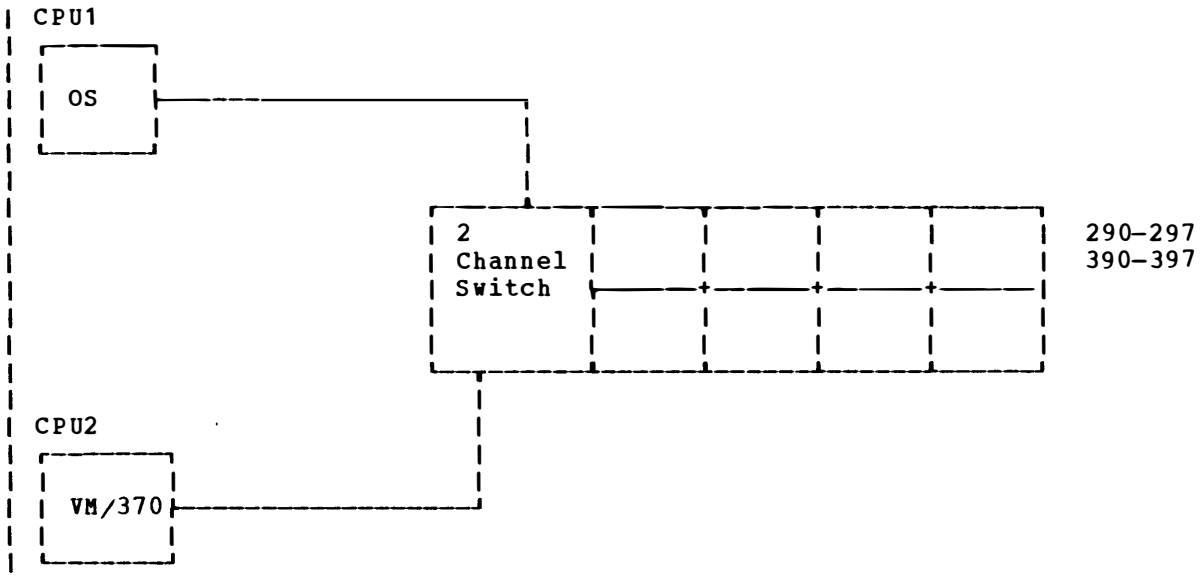
VM/370 USING CHANNEL SWITCHING

VM/370 does not itself include any channel switching facilities. However, channel switching can be used between VM/370 and another operating system if the other operating system supports channel switching. The two- or four-channel switch can be used between:

- Two CPUs, one running VM/370 and the other running an operating system that supports channel switching.
- VM/370 and an operating system running in a virtual machine under the control of VM/370 on one CPU, if the virtual machine operating system supports channel switching.

CHANNEL SWITCHING BETWEEN TWO CPUS

You can use the two- or four-channel switch for devices attached to two CPUs. For example, one CPU could be running VM/370 and the other could be running OS. The configuration is:



VM/370 requires the following RDEVICE and RCTLUNIT macros to support this configuration:

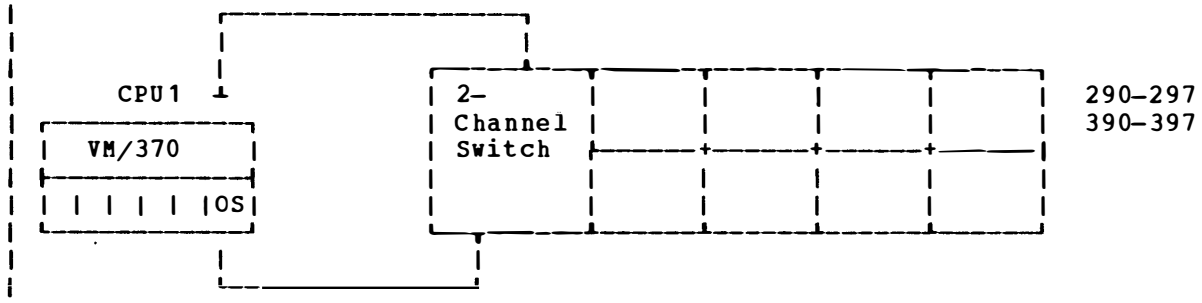
```
      RDEVICE  ADDRESS=(290,8),DEVTYPE=3330,FEATURE=2CHANSW
      RDEVICE  ADDRESS=(390,8),DEVTYPE=3330,FEATURE=2CHANSW
      RCTLUNIT ADDRESS=290,CUTYPE=3830
      RCTLUNIT ADDRESS=390,CUTYPE=3830
```

These macros make it possible for you to run VM/370 on CPU1 or CPU2. If you are always going to run VM/370 on CPU2, you can eliminate one path (eliminate one RDEVICE and RCTLUINT set of macros).

If any I/O devices controlled by VM/370 for its own exclusive use are attached to a control unit with a two- or four-channel switch, the CPU controlling the other channel interface must vary the CP-owned devices offline. For example, if all eight disks in the preceding configuration are mounted and two of those disk are CP-owned volumes (such as CP system residence and CP paging and spooling volumes), the OS system running on CPU1 must vary the CP-owned volumes offline. This procedure protects volumes that CP needs.

CHANNEL SWITCHING ON ONE CPU

You can also use the two- or four-channel switch for devices attached to one CPU that is running VM/370. For example, one CPU could be running VM/370 with OS running in a VM/370 virtual machine. The configuration is:



VM/370 requires the following RDEVICE and RCTLUNIT macros to support this configuration:

```
RDEVICE  ADDRESS=(290,8),DEVTYPE=2314,FEATURE=2CHANSW
RDEVICE  ADDRESS=(390,8),DEVTYPE=2314,FEATURE=2CHANSW
RCTLUNIT ADDRESS=290,CUTYPE=IFA
RCTLUNIT ADDRESS=390,CUTYPE=IFA
```

For this example, you should have all the devices associated with one path offline when you load VM/370. Otherwise, the following message is displayed:

```
DMKCPI954E DASD raddr VOLID volid NOT MOUNTED,
           DUPLICATE OF DASD raddr
```

In this example, the 2314 DASD devices can only be used by the OS system running in a virtual machine if they are dedicated to that virtual machine via the ATTACH command or the DEDICATE control statement in the VM/370 directory. The device addresses generated for the virtual machine operating system do not need to be the same as those defined for the real machine. Also, any operating system that is running under the control of VM/370, and using channel switching with VM/370, cannot access minidisks which are on devices with channel switching. Because reserve/release is supported only for minidisks that are full packs, minidisks that are not defined as full packs cannot be accessed via channel switching on one CPU.

However, minidisks can be used if one of the channel interfaces is using the device as a dedicated device and the other channel interface is using reserve/release support. In this example, such support implies a full pack minidisk. Suppose that the channel 2 devices are online and channel 3 device offline when VM/370 is loaded. Also suppose that 291 contains a full pack minidisk. When VM/370 is loaded, device 291 is considered a CP system device. VM/370 virtual machines can access the full pack minidisk at 291, and it has reserve/release protection. Then, the OS virtual machine controlling the channel 3 interface can have the same pack dedicated to it as 391. The OS virtual machine can only access the full pack minidisks as 391; other VM/370 virtual machines can only access it as 291.

As another example, consider channel switching for tapes. If the real configuration includes a 2816 Switching Unit or a Two- or Four-Channel Switch feature, it can be made to operate under control of a virtual machine operating system. For example, if 580 and 680 are the alternate device addresses for a particular tape drive, then:

- Generate the virtual machine operating system for the appropriate hardware (in this case a 2816 Switching Unit on channels 5 and 6).

- Generate CP as though 580 and 680 are different devices (with different control units and channels) and generate them with the Two-Channel Switch feature (FEATURE=2CHANSW).

- Issue the CP ATTACH command for both device addresses (580 and 680) whenever the real device is to be attached to the virtual machine.

The device addresses generated for the virtual machine operating system do not need to be the same as those on the real machine.

The devices must be used by the virtual machine as dedicated devices (attached, or defined with a DEDICATE statement in the VM/370 directory).

OPERATING SYSTEMS USING RESERVE/RELEASE

VM/370 does not include any reserve/release support for virtual machines sharing DASD volumes. If the real configuration includes a Two- or Four-Channel Switch feature, reserve/release support can be made to operate under control of the virtual machine operating system. In this way, a virtual machine can share a DASD volume (either virtual or real).

For example, if 230 and 330 are alternate device addresses for a particular DASD facility to be shared by USERA and USERB (two virtual machines running on the same real computing system), then:

- Generate the virtual machine operating system for USERA to support the appropriate hardware (device at 230, Two-Channel Switch) and software (RESERVE/RELEASE).

- Generate the virtual machine operating system for USERB to support the appropriate hardware (device at 330, Two-Channel Switch) and software (RESERVE/RELEASE).

- Generate CP as though 230 and 330 were different devices (with different control units and channels); and generate them with the Two-Channel Switch feature (FEATURE=2CHANSW).

- Issue the CP ATTACH command, attaching device 230 to USERA and device 330 to USERB.

If the system generated for USERB is to run in a real machine rather than a virtual machine:

- Generate the CP system with device 230 but not 330.

- Issue the CP ATTACH command, attaching device 230 to USERA.

In both cases:

- The device addressess generated for systems to run in a virtual machine do not need to be the same as on the real machine.

- The devices used by virtual machines must be dedicated (attached or defined with a DEDICATE record in the VM/370 directory). Minidisks cannot be shared in this manner.

Planning Considerations for Remote 3270s

VM/370 supports 3270 display units and printers attached to binary synchronous communications lines. Both remote cluster and standalone configurations are supported. This support includes:

- Nonswitched point-to-point binary synchronous transmission.
- A maximum of 32 display stations and/or printers, in any combination, attached to each 3271 control unit.
- A remote 3270 copy function.
- EBCDIC (Extended Binary Coded Decimal Interchange Code) transmission code only.
- Remote 3270 terminals supported as virtual machine operator consoles.
- CP commands allowing the operator to activate and deactivate the teleprocessing lines and remote stations.
- The CMS Editor support for remote 3270s.
- The recording of MDR (Miscellaneous Data Recorder) records and OBR (Outboard Recording) records on the VM/370 error recording cylinder. The MDR records are for the station and the OBR records are for the line. The CPEREP program edits and prints these records.

The 3270 hard-copy support is limited to remote 3270 virtual machine consoles. You assign a screen copy function to a 3270 program function key. Depressing that key transfers the current display image, in its entirety, to an available printer attached to the same control unit. If the printer is busy when the copy function is invoked, the virtual machine operator receives a NOT ACCEPTED message in the screen's status area.

The following restrictions apply to VM/370's support of remote 3270s:

- Remote 3270 terminals cannot be used as primary or alternate VM/370 system consoles.
- The number of binary synchronous lines supported by VM/370 for remote 3270 use is 16 minus the number of 3704/3705 Communications Controllers in NCP mode minus one (if there are any 3704/3705 Communications Controllers in EP mode).

3270 SUPPORT ON BINARY SYNCHRONOUS LINES

The supported display devices on binary synchronous lines have the same flexibility and usefulness as locally attached 3270 devices, except for the following limitations:

- Display Information Inquiry and Retrieval Speed -- Because the 3270 remote stations are subject to (1) relatively slow teleprocessing transmission speeds and (2) the mechanics of polling operations,

screen display and data entry are not as rapid for remote 3270 devices as they are for locally attached 3270s.

- CP DIAL and ATTACH Commands -- The CP DIAL and ATTACH commands are not supported for remote 3270 stations.

- Hard Copy of 3270 Screen Image -- Just as users of locally attached 3270 display devices can spool their virtual console input and output to the system printer, so can the users of remote 3270 display devices. However, for remote 3270 users, the terminal is physically distant from the system printer and the printer's output is not readily accessible. To improve accessibility, VM/370 provides, within its 3270 remote support, a limited hard-copy facility at the remote location.

- Test Request Key -- The Test Request key on the 3270 terminal is not supported for remote 3270s.

HARDWARE CONFIGURATION SUPPORTED

VM/370's support of remote 3270s requires:

- A binary synchronous line
- A control unit
- Terminal devices (display stations and/or printers)

The binary synchronous line must be in 2701/2703 mode. The transmission control units supporting remote 3270s on binary synchronous lines are:

- Integrated Communications Adapter (ICA).

- IBM 2701 Data Adapter Unit with Synchronous Data Adapter Type II.

- IBM 2703 Transmission Control with Synchronous Data Adapter Type II.

- IBM 3704 and 3705 Communications Controllers in emulation mode. A 3704/3705 line is in emulation mode when it is controlled by the Emulation Program (EP) or the Partitioned Emulation Program (PEP) in EP mode.

Cluster and standalone control units are supported for remote 3270s. The IBM 3271 Control Unit, Model 2, supports clusters of up to 32 remote display stations and/or printers. The IBM 3275 Display Station, Model 2, is the standalone 3270 device that can be attached remotely.

The following devices can be attached to the 3271 control unit:

- IBM 3277 Display Station, Model 2
- IBM 3284 Printer, Model 2
- IBM 3286 Printer, Model 2
- IBM 3288 Line Printer, Model 2

The IBM 3275 Display Station, Model 2, is a standalone control unit and display station. You can attach the IBM 3284 Printer, Model 3, to the 3275. In addition, you can attach the IBM 3286 Printer, Model 3, to the 3275 if RPQ MB4317 is installed.

SYSTEM GENERATION REQUIREMENTS

When you generate VM/370 you must code the appropriate CLUSTER, TERMINAL, and RDEVICE macros and assemble them as part of the DMKRIO (real I/O configuration) file. Then, after the DMKRIO file assembles successfully, you must make a list of the resource identification codes of all the remote 3270 lines and terminals. Give the list to the operations group at your installation; the members of that group need this information when they issue the CP commands that control the operation of remote 3270 lines and devices.

THE CLUSTER MACRO

Code one CLUSTER macro for each 3271 control unit and each 3275 display station. Only a maximum of 16 CLUSTER macros is allowed. Each CLUSTER macro must have a unique label. The CLUSTER macro is described in the "Preparing the Real I/O Configuration File (DMKRIO)" section of "Part 2: Defining Your VM/370 System."

THE TERMINAL MACRO

Code one TERMINAL macro for each display station and printer that is attached to the clustered 3271 control unit.

Code one TERMINAL macro for the 3275 display station. If a 3284 or 3286 printer is attached to the 3275, code MODEL=3 on the TERMINAL macro. The TERMINAL macro is described in the "Preparing the Real I/O Configuration File (DMKRIO)" section of "Part 2: Defining Your VM/370 System."

THE RDEVICE MACRO

Code one RDEVICE macro for each binary synchronous line used for remote 3270s (code one RDEVICE macro for each CLUSTER macro you code). A maximum of 16 RDEVICE macros for lines to support remote 3270s is allowed.

The RDEVICE macro is described in Part 2. However, the format of the RDEVICE macro for the binary synchronous line and transmission control unit for remote 3270s is included here to help you code the macro correctly. The format of an RDEVICE macro for a communication line that supports remote 3270s is:

Name	Operation	Operands
	RDEVICE	ADDRESS=cuu,
		RDEVICE={2701, 2703, ICA, 3704, 3705},
		ADAPTER=BSCA
		[,BASEADD=cuu]
		,CLUSTER=label

where:

ADDRESS=cuu
is the real I/O address of the binary synchronous line to be used by remote 3270s.

The address, cuu, is three hexadecimal digits from 000 to FFF.

DEVTYPE={2701, 2703, ICA, 3704, 3705}
is the device type of the transmission control unit that controls the line.

Note: The lines attached to the 3704/3705 must be controlled by the Emulation Program (EP) or the Partitioned Emulation Program (PEP) in EP mode.

ADAPTER=BSCA
is the terminal adapter for the transmission control unit. BSCA represents the:

- IBM Binary Synchronous Terminal Adapter Type II for a 2701
- IBM Binary Synchronous Terminal Control Type II for a 2703
- Integrated Communications Adapter (ICA) on the System/370 Model 135
- IBM 3704 and 3705 Communications Controllers

BASEADD=cuu
is the native subchannel address (load address) of the 3704/3705 that controls the physical line or lines. This operand is required for correct operation of VM/370 recovery management for emulator lines on a 3704/3705. Specify this operand only for 3704 and 3705.

CLUSTER=label
is the label of a CLUSTER macro that defines the cluster or standalone station attached to this line.

| Examples

| The following examples are RDEVICE macros describing a nonswitched | point-to-point communication line connected to a 2701, 2703, and 3705.

| Example 1:

```
|       RDEVICE ADDRESS=078,DEVTYPE=2701,ADAPTER=BSCA, CLUSTER=CLUST001
```

| The cluster station that is connected to this line is defined by the | CLUSTER macro labeled CLUST001. The line at address 078 is controlled | by a 2701 transmission control unit.

| Example 2

```
|       RDEVICE ADDRESS=080,DEVTYPE=2703,ADAPTER=BSCA, CLUSTER=CLUST020
```

| The line at address 080 is controlled by a 2703 transmission control | unit and the corresponding CLUSTER macro is labeled CLUST020.

| Example 3

```
|       RDEVICE ADDRESS=0B8,DEVTYPE=3705,ADAPTER=BSCA,                  X
|               BASEADD=0B0,CLUSTER=CLUST030
```

| The line at address 0B8 is controlled by a 3705 Communications | Controller and the corresponding CLUSTER macro is labeled CLUST030.

| THE RESOURCE IDENTIFICATION CODES

| After the real I/O configuration file (DMKRIO) assembles successfully, | generate a list of the resource identification codes that correspond to | each line address. Give the list to the operations group at your | installation. The operator needs to know the resource identification | code when he issues the commands to control the operation of the remote | 3270 lines and terminals.

| The resource identification code is a four-digit hexadecimal code. | The low-order three digits of the resource identification code are the | resource address. The high-order digit is the line code.

| The resource address is generated by VM/370; the order in which the | TERMINAL macros appear in the real I/O configuration file (DMKRIO) | determines the resource addresses of the terminals defined. Each | CLUSTER macro defines a 3270 control unit; each 3270 control unit has a | resource address of X'00'. The device defined by the first TERMINAL | macro after the CLUSTER macro (in the DMKRIO file) has a resource | address of X'01', the second has a resource address of X'02', up to the | maximum of X'20'. This resource address makes up the low-order three | digits of the resource identification code.

| The line code is also generated by VM/370. You have to refer to the | assembly listing for DMKRIO to determine the line code (the high-order | digit of the resource identification code). Locate the label DMKRIORN | near the end of the DMKRIO assembly listing. This label identifies a | list of all the lines used by remote 3270s and by 3704/3705 | Communications Controllers in NCP mode. The high-order digit is the | line code and is assigned according to the order in which the line | addresses appear in the list. The first line address is assigned a line | code 0 to complete its resource identification code, the second is | assigned 1, and so on up to the last line. VM/370 supports a maximum of

Remote 3270s

16 binary synchronous lines for use by remote 3270s, thus the maximum value of the high-order digit is F. Figure 13.1 shows you a sample DMKRIO assembly listing and the corresponding line codes.

Sample of DMKRIO Assembly Listing			Line Code (in hexadecimal)
DMKRIORN	DC	F'4'	
	DC	AL2((RDV078-DMKRIODV)/8)	
	DC	XL2'078'	0
	DC	AL2((RDV07A-DMKRIODV)/8)	
	DC	XL2'07A'	1
	DC	AL2((RDV079-DMKRIODV)/8)	
	DC	XL2'079'	2
	DC	AL2((RDV07B-DMKRIODV)/8)	
	DC	XL2'07B'	3

Figure 13.1. Example of Determining Line Code for Remote 3270 Resource Identification Codes

Once you determine the resource identification codes for the devices in your remote 3270 configuration, generate a list for operations. The list should include the following information:

- Line address
- Line code
- Resource address
- Label of plug on control unit panel
- Resource Identification Code
- Device type

Note: The plug panel of the 3271 control unit has up to 32 ports where terminals can be attached. Each of these ports is labeled. The first port is labeled 0, the second is labeled 2, and so on up to 31.

AN EXAMPLE OF A REMOTE 3270 CONFIGURATION

This example shows you the contents of the real I/O configuration file to define the following remote 3270 configuration:

- A clustered 3271 control unit with eight ports
- A standalone 3275 display station

The macros are coded so that the 3271 clustered control unit can support eight display devices, or six display devices and two printers. To define such a configuration, you must code 2 CLUSTER, 16 TERMINAL, and 2 RDEVICE macros defining the 2 separate clusters. A 3275 standalone control unit, with one display and one printer, is also supported by the following macros. To define it, you must code one CLUSTER, one TERMINAL, and one RDEVICE macro.

The real I/O configuration file for this example is:

```
DMKRIO    CSECT
CLUST078  CLUSTER   CUTYPE=3271,GPOLL=407F,LINE=078
          TERMINAL  TERM=3277,SELECT=6040,FEATURE=OPRDR,MODEL=2
          TERMINAL  TERM=3277,SELECT=60C1,FEATURE=OPRDR,MODEL=2
          TERMINAL  TERM=3277,SELECT=60C2,FEATURE=OPRDR,MODEL=2
          TERMINAL  TERM=3277,SELECT=60C3,FEATURE=OPRDR,MODEL=2
          TERMINAL  TERM=3277,SELECT=60C4,FEATURE=OPRDR,MODEL=2
          TERMINAL  TERM=3277,SELECT=60C5,FEATURE=OPRDR,MODEL=2
          TERMINAL  TERM=3286,SELECT=60C6,MODEL=2
          TERMINAL  TERM=3284,SELECT=60C7,MODEL=2
CLUST07A  CLUSTER   CUTYPE=3275,GPOLL=407F,LINE=07A
          TERMINAL  TERM=3275,SELECT=6040,FEATURE=OPRDR,MODEL=3
CLUST079  CLUSTER   CUTYPE=3271,GPOLL=407F,LINE=079
          TERMINAL  TERM=3277,SELECT=6040,FEATURE=OPRDR,MODEL=2
          TERMINAL  TERM=3277,SELECT=60C1,FEATURE=OPRDR,MODEL=2
          TERMINAL  TERM=3277,SELECT=60C2,FEATURE=OPRDR,MODEL=2
          TERMINAL  TERM=3277,SELECT=60C3,FEATURE=OPRDR,MODEL=2
          TERMINAL  TERM=3277,SELECT=60C4,FEATURE=OPRDR,MODEL=2
          TERMINAL  TERM=3277,SELECT=60C5,FEATURE=OPRDR,MODEL=2
          TERMINAL  TERM=3277,SELECT=60C6,FEATURE=OPRDR,MODEL=2
          TERMINAL  TERM=3277,SELECT=60C7,FEATURE=OPRDR,MODEL=2
          RDEVICE   ADDRESS=078,DEVTYPE=3705,ADAPTER=BSCA,BASEADD=0B0, X
                    CLUSTER=CLUST078
          RDEVICE   ADDRESS=07A,DEVTYPE=3705,ADAPTER=BSCA,BASEADD=0B0, X
                    CLUSTER=CLUST07A
          RDEVICE   ADDRESS=079,DEVTYPE=3705,ADAPTER=BSCA,BASEADD=0B0, X
                    CLUSTER=CLUST079
```

In this configuration, if the 3271 cluster control unit is on line 078 there are six display devices and two printers supported. If the 3271 cluster control unit is on line 079, eight display devices and no printers are supported. Display devices can be interchanged among resource addresses allocated to display devices and printers can be interchanged among resource addresses allocated to printers; but a printer cannot be attached at an address defined for a display device, and vice versa.

After the DMKRIO file assembles successfully, you should generate a list of resources for the operations group. Your list should be similar to the list shown in Figure 13.2; it corresponds to the preceding example.

Line Address	Line Code	Resource Address	Label of Plug in Control Unit Panel	Resource Identification Code	Device Type
078	0	000	--	0000	cluster
		001	0	0001	display
		002	1	0002	display
		003	2	0003	display
		004	3	0004	display
		005	4	0005	display
		006	5	0006	display
		007	6	0007	printer
		008	7	0008	printer
079	2	000	--	2000	cluster
		001	0	2001	display
		002	1	2002	display
		003	2	2003	display
		004	3	2004	display
		005	4	2005	display
		006	5	2006	display
		007	6	2007	display
		008	7	2008	display
07A	1	000	--	1000	cluster
		001	--	1001	display
		002	--	1002	printer

Note: The line code is determined by refering to Figure 13.1; it corresponds to this example.

Figure 13.2. Sample List of Resource Identification Codes for Operations

Generating a VM/370 System that Supports the 3704/3705

The generation of a 3704 or 3705 Communications Controller control program that runs under the control of VM/370 is normally done after the VM/370 system generation is completed. However, when a 3704 or 3705 is to be generated, the following preparations must be made:

- An RDEVICE macro instruction for the 3704 or 3705 must be included in the real I/O configuration (DMKRIO) file.

- 3704/3705 control programs that are to be used by VM/370 must be stored on a CP-owned volume in the page format that is currently used for saved virtual machine systems (that is, those created by the SAVESYS command). Each 3704/3705 control program image to be saved must be defined by a NAMENCP macro instruction in the system name table (CP module DMKSNT) and saved with the SAVENCP command.

- Enough space to contain the 3704/3705 control program image must be allocated on the CP-owned volume specified in the NAMENCP macro instruction.

| Note: The alternate console for VM/370 must not be on a telecommunications line on a real 3704/3705, unless the 3704/3705 is loaded by another operating system (OS/VS1, OS/VS2, or DOS/VS) before VM/370 is loaded.

Part 4 contains a complete discussion on generating a 3704 or 3705 control program. It describes the support provided with NCP, EP, and PEP control programs and tells you how to generate the 3704/3705 control program, step by step.

CODING THE RDEVICE MACRO

The RDEVICE macro is described in Part 2. However, the format of the RDEVICE macro for a 3704/3705 is included here to help you code the macro correctly. The format of the RDEVICE macro for an IBM 3704/3705 is:

```
Name   Operation   Operands
       RDEVICE     ADDRESS={ cuu     },
                           { (cuu,nn) },

                   DEVTYPE={3704}
                           {3705},

                   ADAPTER={TYPE1}
                           {TYPE2}
                           {IBM1 }
                           {TELE2}
                           {BSCA }

                   [,MODEL={ 1 }  [,SETADDR={ 0 }]
                           { 2 }            { 1 }
                           { 3 }            { 2 }
                           { 4 }            { 3 }
                           { 5 }            { 4 }
                           { 6 }
                           { 7 }
                           { 8 }

                   [,CPTYPE={EP }  [,CPNAME=ncpname]
                           {NCP}
                           {PEP}

                   [,MAXDIAL=nn] [,BASEADD=ccu]
```

where:

ADDRESS={cuu / (cuu,nn)}

is the real device address (cuu) of the 3704/3705. Use the (cuu,nn) form to generate multiple (nn) real device blocks (RDEVBLOKs) when CPTYPE=PEP (or EP) is specified.

DEVTYPE={3704 / 3705}

is the device type. You should specify 3704 or 3705 instead of 2701, 2702, or 2703 when CPTYPE = PEP or EP.

ADAPTER={TYPE1 / TYPE2 / IBM1 / TELE2 / BSCA}

identifies either the channel adapter accessed by the specified real address (TYPE1 or TYPE2), or a line adapter if this is an emulator line group (IBM1, TELE2, or BSCA).

MODEL={1 / 2 / 3 / 4 / 5 / 6 / 7 / 8}

is the 3704/3705 model number. The model determines the size of the 3704/3705 storage. See Figure 14 for a list of the model numbers and the corresponding storage sizes.

IBM 3704 Communications Controller		IBM 3705 Communications Controller	
Model	Storage	Model	Storage
A1	16K	A1, B1, C1, D1	16K
A2	32K	A2, B2, C2, D2	48K
A3	48K	B3, C3, D3	80K
A4	64K	B4, C4, D4	112K
		C5, D5	144K
		C6, D6	176K
		D7	208K
		D8	240K

Figure 14. IBM 3704/3705 Models

SETADDR={0 | 1 | 2 | 3 | 4} indicates which set address (SAD) command must be issued prior to enabling the emulator lines.

CPTYPE={EP | NCP | PEP} indicates which type of 3704/3705 control program is run in this 3704/3705. If any of the three may be run, omit the CPTYPE operand, or specify CPTYPE=EP. If CPTYPE=NCP is specified, an emulator program may not load successfully.

CPNAME=ncpname is the one- to eight-character name of the control program to be automatically loaded in the 3704/3705 at system IPL time. If an automatic load is not desired, omit this operand.

MAXDIAL=nn is the maximum number of dynamically specified 2701/2702/2703 emulator lines which may be active at any one time. "nn" additional RDEVBLOKs are generated to accommodate the emulation lines.

This operand is valid only if CPTYPE=PEP.

BASEADD=ccu is the native address (load address) of the 3704/3705 which controls the physical line(s). This operand is required for correct operation of VM/370 recovery management for emulator lines based on a 3704/3705.

This operand is valid only if ADAPTER=IBM1, TELE2, or BSCA.

SPECIAL CONSIDERATIONS FOR CODING THE RDEVICE MACRO

The 3704/3705 Communications Controllers have many varied uses. Consequently, the control program generation for the 3704/3705 is complex.

EP-Only Control Programs

If the 3704/3705 is to be run in emulation mode:

- Use the (cuu,nn) form of the ADDRESS operand to generate multiple RDEVBLOKs.
- Omit the CPTYPE and MAXDIAL operands.
- Specify the appropriate name for CPNAME.

To generate additional emulator lines for the same 3704/3705, use the following coding guidelines on subsequent RDEVICE macros:

- Omit the CPTYPE, CPNAME, MAXDIAL, and MODEL operands.
- Specify the ADAPTER as IBM1, TELE2, or BSCA, as appropriate.

For ADAPTER=IBM1 (or TELE2), the SETADDR operand must also be specified, exactly as if the device were a 2702 or 2703. Special care must be taken to ensure that the CPNAME and ADAPTER operands appear only in one RDEVICE macro for each physical 3704/3705.

Note: If you use the (cuu,nn) form of the ADDRESS operand to generate multiple RDEVBLOKs and also specify the CPNAME and ADAPTER=TYPE1 operands on the RDEVICE macro, the additional RDEVBLOKs are generated as ADAPTER=IBM1 and SETADDR=4.

For NCP-Only Control Programs

If the 3704/3705 is only to be used with the NCP:

- Specify CPTYPE=NCP.
- Omit the MAXDIAL and SETADDR operands.
- Specify either ADAPTER=TYPE1 (or TYPE2).

This prohibits the use of virtual 2701/2702/2703 lines via the 3704/3705 (unless separate RDEVICE macros are used to generate additional RDEVBLOKs). In addition, be sure that the CPNAME and ADAPTER operands appear only in one RDEVICE macro for each physical 3704/3705.

For PEP Control Programs

If CPTYPE=PEP is specified, the necessary RDEVICE operands depend on the manner in which the PEP is to be used:

- If all of the 3704/3705 lines are specified as initially in NCP mode (USE=NCP on the LINE macro in the 3704/3705 Stage 1 input file, or USE operand omitted), the MAXDIAL operand must be used to generate additional RDEVBLOKs. The number to be generated is a consideration of fixed storage requirements (each RDEVBLOK requires 80 bytes of fixed storage) and the 2701/2702/2703 capabilities. The number specified for MAXDIAL is an absolute limit on the number of 2701/2702/2703 emulation lines which can be active at any one time, regardless of how many lines are specified as TYPE=PEP in the 3704/3705 control program generation.

- If no lines were specified as TYPE=PEP in the 3704/3705 generation (that is, all lines are specified as either TYPE=NCP or EP), use the (cuu,nn) form to define the RDEVBLOKs for the emulator lines, and omit the MAXDIAL operand.

  Note: If you use the (cuu,nn) form of the ADDRESS operand to generate multiple RDEVBLOKs and also specify the CPNAME and ADAPTER=TYPE1 operands on the RDEVICE macro, the additional RDEVBLOKs are generated as ADAPTER=IBM1 and SETADDR=4.

- If the 3704/3705 generation includes a mixture of the NCP and PEP lines (that is, the corresponding LINE macros have a combination of TYPE=NCP, EP, and PEP), use the ADDRESS operand to generate emulator-only lines and use the MAXDIAL operand to specify a limit on how many PEP lines can be varied to emulation mode (these are in addition to the emulator-only lines).

- If you specify TYPE=PEP and USE=EP, the lines generated in this fashion cannot be used as NCP lines by VM/370 without explicit operator action.

Summary of RDEVICE Macro Coding Considerations

For each physical 3704/3705, there should be only one RDEVICE macro which specifies the ADAPTER=TYPE1 or TYPE2, MODEL, CPTYPE, MAXDIAL, and CPNAME operands. This RDEVICE macro defines the base address of the 3704/3705 (that is, the real address used to perform the load and dump operations). If the physical device is a 3705 with two channel adapters installed, there may be a second RDEVICE macro which specifies the ADAPTER=TYPE1 or TYPE2, MODEL, and CPTYPE operands. There must never be a second use of the CPNAME operand; the MAXDIAL operand need not be coded twice since a 3705 cannot have two TYPE1 channel adapters. Even if CPTYPE=EP is specified, the 3704/3705 base address cannot be used as a telecommunications line; its function is only to load and dump the 3704/3705, and the device type and class are different from those of all other lines generated.

Whenever there is more than one subchannel address (CPTYPE=EP or PEP), include in the DMKRIO deck all of the RCTLUNIT macros required to specify those real addresses which the EP or PEP control program may use. For example, assume all of the lines on a 3704/3705 are specified as TYPE=PEP, the MAXDIAL number is specified as 10, and the real addresses for the 3704/3705 are 020 (base address) and 021-07F. In this case, a RCTLUNIT macro must be included to cover the device address range 020-07F, even though no more than ten emulator lines could be active at one time.

If you have a 3704/3705 and a 2701/2702/2703 on the same VM/370 system, the virtual addresses for the 3704/3705 must not be the same as any of the real 2701/2702/2703 addresses.

Examples

Examples of RDEVICE macro specifications follow. For convenience, the continuation character in position 72 is not shown.

Example 1

```
RDEVICE ADDRESS=320,
        DEVTYPE=3705,
        MODEL=4,
        ADAPTER=TYPE2,
        CPTYPE=NCP,
        CPNAME=NCP320
```

This describes a 3705 Model B4 (112K storage) at address X'320', with a type two channel adapter. The 3705 is to be automatically loaded with the Network Control Program 'NCP320'.

Example 2

```
RDEVICE ADDRESS=(020,16),
        DEVTYPE=3704,
        MODEL=2,
        ADAPTER=TYPE1,
        CPNAME=CEP020
```

This describes a 32K 3704 at address X'020', with 15 emulator lines at addresses X'021' to X'02F'. The 3704 is to be loaded with the Emulation Program 'CEP020'.

Example 3

```
RDEVICE ADDRESS=(030,16),
        DEVTYPE=3704,
        ADAPTER=IBM1,
        SETADDR=2,
        BASEADD=020
```

This describes an additional 16 emulator lines on the same 3704 specified by Example 2.

Example 4

```
RDEVICE ADDRESS=0B0,
        DEVTYPE=3705,
        MODEL=4,
        ADAPTER=TYPE1,
        CPTYPE=PEP,
        CPNAME=PEP0B0,
        MAXDIAL=16

RDEVICE ADDRESS=(0B1,15),
        DEVTYPE=3705,
        ADAPTER=BSCA,
        BASEADD=0B0
```

These two macros describe a 112K 3705 for use with the Partitioned Emulation Program 'PEP0B0', at base address X'0B0', with a maximum of 16 start/stop emulator lines (switched mode), and 15 binary synchronous lines (always in emulator mode) at addresses X'0B1' to X'0BF'.

CREATING AN ENTRY IN THE SYSTEM NAME TABLE

You must create an entry in the system name table (DMKSNT) for each unique 3704/3705 control program that you generate. If you can foresee generating several versions of the 3704/3705 control program, define extra entries in the system name table when you generate VM/370. In this way, you do not have to regenerate the VM/370 system just to update the system name table. If you should have to regenerate the VM/370 system just to add a new entry to the system name table, see the discussion about the GENERATE EXEC procedure in "Part 6: Updating VM/370."

The NAMENCP macro is described in Part 2.

RESERVING DASD SPACE FOR THE 3704/3705 CONTROL PROGRAM IMAGE

DASD space to contain the 3704/3705 control program image must be reserved on a CP-owned volume. The DASD space reserved should be sufficient to contain the number of pages specified in the SYSPGCT operand of the NAMENCP macro, plus one or more for system use, as follows:

- If CPTYPE=EP, allow only one extra page.
- If CPTYPE=NCP or PEP, allow one extra page for the first 1000 resource IDs defined, and one additional page for each additional 1000. The number of resource IDs defined can be determined from the output listing of the 'fname01' Stage 2 assembly. The extra page requirements never exceed a total of four pages.

These additional pages are used to store the reference table information provided by the SAVENCP program. The page-record capacity for one cylinder of DASD space on differing device types is in the "Formatting Volumes -- General Information" section of Appendix F.

Saved Systems

Saved systems are described in detail in the VM/370: System Programmer's Guide. If you plan to save core-image copies of virtual machine operating systems you should do the following when you generate VM/370.

- Create an entry in the system name table for each system you wish to save.

- Reserve space on a CP-owned volume for each system you wish to save.

You create entries in the system name table by coding NAMESYS and NAMENCP macros and assembling the system name table (DMKSNT) file during system generation. You specify which volumes are to be owned by CP by coding the SYSOWN macro and assembling the CP system control (DMKSYS) file during system generation. These macros and files are described in Part 2.

If you decide to add entries to the system name table after you have installed VM/370, you must code the appropriate NAMESYS or NAMENCP macros, reassemble the system name table module (DMKSNT), and reload the CP nucleus. Likewise, if you must add a CP-owned volume after system generation, you must recode the SYSOWN macro, reassemble the CP system control module (DMKSYS), and reload the CP nucleus. Use the GENERATE EXEC procedure to reassemble DMKSNT and/or DMKSYS and to reload the CP nucleus. GENERATE is described in "Part 6: Updating VM/370."

Estimating VM/370 Storage Requirements

This section contains information about:

- Estimating real storage requirements for VM/370
- Reducing the size of the CP nucleus
- Estimating direct access storage requirements
- Estimating storage requirements for CMS minidisks

The "Specifying a Virtual=Real Machine" section includes information about estimating real storage requirements for a virtual=real machine.

REAL STORAGE REQUIREMENTS FOR CP

The following chart lists the various CP requirements and the amount of real storage required for each.

CP Requirement	Real Storage Allocated
Resident nucleus	Approximately 132K
Internal trace table	4K of storage is allocated for each 256K of real storage. This storage is set aside at IPL time.
Real control blocks	There is a control block for each real device, control unit, and channel: • 80 bytes/real device • 64 bytes/real control unit • 96 bytes/real channel • 24 bytes for each remote 3270 or real 3704/3705
Permanently allocated free storage (virtual control blocks and tables)	A minimum of 12K, plus an additional 4K for each 64K of real storage above 256K. This storage is set aside at IPL time. Each logged-on virtual machine requires a virtual machine control block (VMBLOK), a segment table (SEGTABLE), a page table (PAGTABLE), a swap table (SWPTABLE), and a control block for each virtual device, control unit, and channel. The storage required is: • 400 bytes for the VMBLOK • 64 bytes/1M of virtual storage for the SEGTABLE • 40 bytes/64K of virtual storage for the PAGTABLE • 136 bytes/64K of virtual storage for the SWPTABLE • 56 bytes/virtual device • 40 bytes/virtual control unit • 40 bytes/virtual channel

For example, if you have:

1M of real storage
29 real devices
6 real control units
3 real channels

and 12 virtual machines defined, each with:

1 virtual reader
1 virtual printer
1 virtual punch
3 virtual disks
3 virtual channels
1 virtual machine console
3 virtual control units
320K of virtual storage

you would estimate CP real storage requirements as follows.

| 132K for the CP resident nucleus
16K for the CP internal trace table
3K for the real control blocks, calculated as follows:

80 X 29 = 2320 bytes for the real devices

64 X 6 = 384 bytes for the real control units

96 X 3 = 288 bytes for the real channels

the sum is: 2320 + 384 + 288 = 2992 bytes (approximately 3K)

60K for permanently allocated free storage

| 211K real storage required

Also, as each of the 12 virtual machines defined logs on, approximately 2K of real storage is allocated to each from the permanently allocated free storage.

400 bytes for a VMBLOK
64 bytes for the SEGTABLE
200 bytes for the PAGTABLE
680 bytes for the SWPTABLE
56 bytes for a virtual reader
56 bytes for a virtual printer
56 bytes for a virtual punch
168 bytes for three virtual disks
120 bytes for three virtual channels
56 bytes for a virtual machine console
120 bytes for three virtual control units

1976 bytes for each of the logged-on users defined

See the "Specifying the Amount of Virtual=Real Space" section for an example of estimating storage requirements and determining the maximum size of the virtual=real area.

REDUCING THE SIZE OF THE CP NUCLEUS

Support for the 3704, 3705, 3066, and 3270 increases the size of the CP nucleus. The 3704/3705 is primarily handled by the module DMKRNH. The graphic device support for locally attached terminals is handled by the module DMKGRF while the remote 3270 support is handled by the DMKRGF and DMKBSC modules. Each of these modules occupies approximately 6000 bytes (decimal) of system nucleus space.

This nucleus area can be reclaimed by deleting DMKRNH, DMKGRF, DMKRGF and DMKBSC from the system loadlist EXEC file. Caution should be exercised before deleting them from the loadlist. If you generate any type of locally attached graphic device in the DMKRIO assemble file, you cannot delete the module DMKGRF. Or, if you generate remote 3270s in the DMKRIO assemble file, you cannot delete the DMKRGF and DMKBSC modules. In addition, if you generate the 3704/3705 in the DMKRIO assemble file, you cannot delete the DMKRNH module.

The following names are undefined during the VMFLOAD procedure if DMKGRF and DMKRNH are deleted from the loadlist:

```
DMKGRFEN    DMKRNHCT    DMKRNHND
DMKGRFIC    DMKRNHIC    DMKRNHTR
DMKGRFIN    DMKRNHIN
```

The following names are undefined during the VMFLOAD procedure if DMKBSC and DMKRGF are deleted from the loadlist:

```
DMKBSCER    DMKRGFIC
DMKRGFEN    DMKRGFIN
```

DIRECT ACCESS STORAGE REQUIREMENTS FOR CP

The following list shows how much DASD space CP requires by DASD type. The following paragraphs describe in detail how you determine the amount of DASD space CP requires for the nucleus, error recording, warm start data, directory, saved systems data, paging and spooling space.

	2314	3330	2305	3340
CP Nucleus	varies	varies	varies	varies
Error Recording	2	2	2	2
Warm Start	1	1	1	1
Directory	2	2	2	2
Saved Systems	varies	varies	varies	varies
Paging Space	32	18	40	40
Spooling Space	50	30		70
Total System[1]	90 cyl	55 cyl	49 cyl	120 cyl

[1]These figures do not include space for saved systems.

CP NUCLEUS DASD REQUIREMENTS

The CP nucleus (without a virtual=real area) currently requires 75 to 80 pages of disk space for resident and pageable functions.

To determine the number of cylinders required for the CP nucleus, refer to the load map produced during system generation. One DASD page is required for each page of fixed and pageable nucleus (for a CP nucleus without a virtual=real area). The formulas for the amount of DASD space needed for a CP nucleus with a virtual=real area are in the "Specifying a Virtual=Real Machine" section.

For example, if the last module entry in the load map is at page 4B (hexadecimal), 75 pages of disk space are required for CP nucleus residence. The number of cylinders required depends on the system residence device used; see the "Saved System DASD Requirements" section that follows for the number of pages per cylinder each device can accommodate.

Normally, the number of cylinders required for CP nucleus residence is:

4 cylinders on a 2305 or 3340
3 cylinders on a 2314 or 2319
2 cylinders on a 3330 or 3333

ERROR RECORDING DASD REQUIREMENTS

Error Recording space is fixed and must be allocated as shown (two cylinders).

WARM START DATA DASD REQUIREMENTS

Formulas for calculating the amount of warm start space needed are in "Part 2: Defining Your VM/370 System" under the discussion of the SYSWRM operand of the SYSRES macro.

VM/370 DIRECTORY DASD REQUIREMENTS

The VM/370 directory normally requires two cylinders so that it can be rewritten without disturbing the active directory and swapped after a successful update. Formulas for computing directory sizes are found in the "Allocating DASD Space for the VM/370 Directory" section of Part 2.

SAVED SYSTEM DASD REQUIREMENTS

Saved systems require one page for each page saved, plus an additional information page. However, a 3704/3705 may require up to four additional information pages.

To save one copy of the CMS system requires two cylinders on a 2314, 2319, or 3340, or one cylinder on a 3330 or 3333.

PAGING AND SPOOLING DASD REQUIREMENTS

Paging and spooling space requirements are installation-dependent. (The values shown in the preceding list are for average systems.) Paging space is allocated at a rate of:

24 pages/cylinder on a 2305 or 3340
32 pages/cylinder on a 2314 or 2319
57 pages/cylinder on a 3330 or 3333
(The 2305 is normally used for paging only.)

Spooling data is placed in a 4K-byte buffer with the necessary channel programs required for each record. Data capacity of spooling cylinders thus varies with the data and CCWs used.

The primary system operator is warned when the paging/spooling space becomes 90% full. The VM/370: Operator's Guide tells the operator what he should do if this warning occurs.

ESTIMATING DASD STORAGE REQUIREMENTS FOR CMS MINIDISKS

The following information is intended to help you allocate sufficient direct access storage space for CMS minidisks.

A 2314 cylinder formatted by the CMS FORMAT command contains 150 800-byte blocks, which can contain approximately 1300 80-byte lines of source programs and data.

A 3330 cylinder formatted by the CMS FORMAT command contains 266 800-byte blocks, which can contain approximately 2300 80-byte lines of source programs and data.

A 3340 cylinder formatted by the CMS FORMAT command contains 96 800-byte blocks, which can contain approximately 960 80-byte lines of source programs and data.

Each 800-byte block contains file control information as well as your data. A given amount of data requires more file information if put into many small files instead of a few large files.

For each an average CMS user, the following minidisk space should be sufficient:

- 7 cylinders of 2314 space (for approximately 9100 80-byte lines of source programs and data)
- 4 cylinders of 3330 space (for approximately 9600 80-byte lines of source programs and data)
- 11 cylinders of 3340 space (for approximately 10560 80-byte lines of source programs and data)

Minidisks

The external storage requirements of multiple virtual machines executing concurrently would be excessive if each virtual machine were assigned one real direct access storage device for each virtual DASD device in its configuration.

Therefore, if you do not require the full capacity of a real DASD device, you can be assigned one or more minidisks instead. A minidisk is a logical subdivision of a physical disk pack with its own virtual device address, virtual cylinders (starting with 0, 1, 2, and so on) and a VTOC (volume table of contents or disk label identifier). Each of your minidisks is preallocated the number of contiguous full cylinders you are apt to require, and that space is considered to be a complete virtual disk device.

Minidisks are controlled and managed by CP. If a virtual machine attempts to use DASD space beyond the boundaries defined for its minidisks, CP presents a command reject (seek check) to the virtual machine. If a system is to be run on both a virtual and a real machine, VM/370 recommends that minidisks not be used for that system unless the minidisk starts at real cylinder zero. For a detailed list of minidisk restrictions, see the VM/370: Introduction.

The remainder of this section describes the following characteristics of minidisks:

- Definition
- Space allocation
- Track characteristics
- Alternate tracks
- Labels

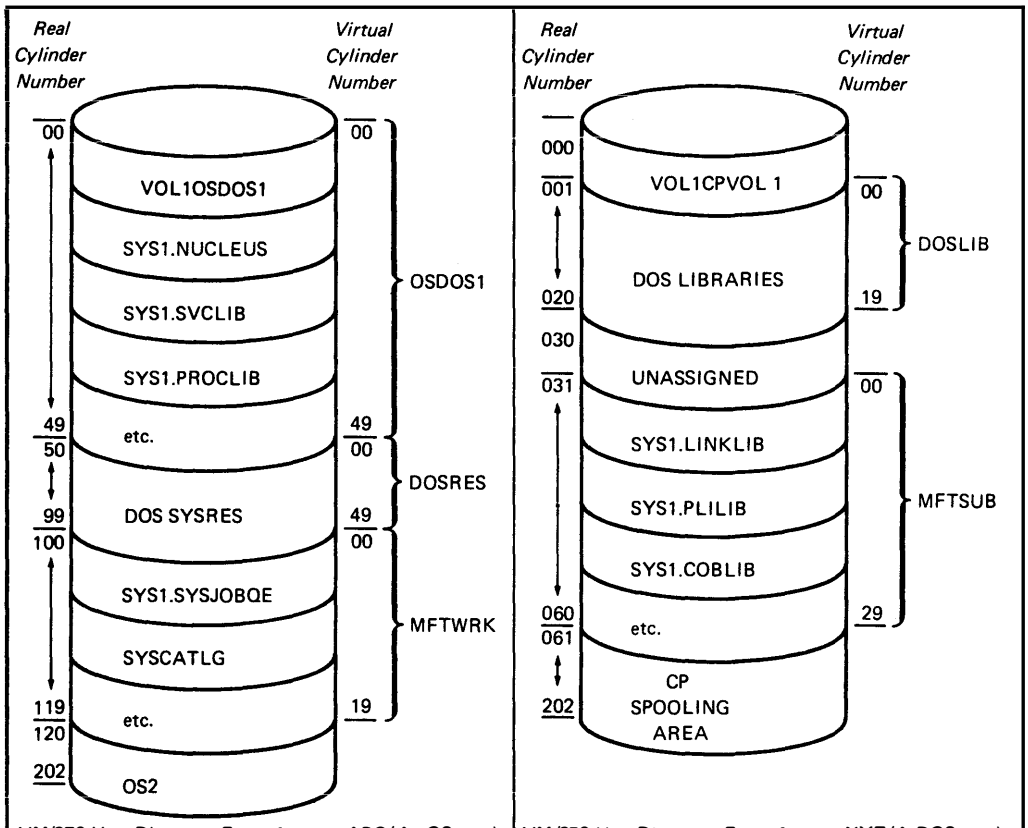
DEFINING MINIDISKS

Permanent minidisks are defined in the VM/370 directory entry for a virtual machine. A minidisk defined in the directory is a permanent part of the virtual machine configuration and the data on the minidisk is available to the user from session to session.

If any virtual machine has a temporary requirement for direct access space, this can be filled from a pool of T-disk space. You specify the size of the T-disk pool when you allocate disk space with the standalone Format/Allocate program. Minidisks created from the T-disk area are available to the virtual machine for the duration of one terminal session. When the virtual machine logs off or issues a CP command to release the temporary minidisk, the area is returned to CP.

It is up to you to allocate minidisks on VM/370 disks in a manner that minimizes arm contention and physical overlap. Information about minimizing arm contention is found in the "Preparing the CP System Control File (DMKSYS)" section of Part 2.

Figure 15 illustrates the use and definition of minidisks. The disk labeled OSDOS1 contains several minidisks, some formatted to OS requirements and others to DOS requirements. OSDOS1 is a 2314 volume. The directory entry for userid ABC (an OS user) describes the virtual device 230 as a 50-cylinder area, and the virtual device 231 as a 20-cylinder area on real volume OSDOS1. The directory entry for userid XYZ (a DOS user) describes the virtual device 130 as a 50-cylinder disk area on a real volume OSDOS1.



VM/370 User Directory Entry for user ABC (An OS user)

```
USER      ABC     123     512K
  ACCOUNT   985
    CONSOLE  009  3215
    MDISK    230  2314  000 050  OSDOS1 W
    MDISK    231  2314  100 020  OSDOS1 W
    MDISK    232  2314  031 030  CPVOL 1 W
    SPOOL    00C  2540  READER A
    SPOOL    00D  2540  PUNCH A
    SPOOL    00E  1403  A
```

VM/370 User Directory Entry for user XYZ (A DOS user)

```
USER  XYZ PASSWORD
   ACCOUNT  NUMBER  BIN14
      CONSOLE   01F   3215
      SPOOL     C     2540 READER
      SPOOL     D     2540 PUNCH
      SPOOL     E     1403
      MDISK     130   2314 050 050  OSDOS1 W
      MDISK     131   2314 001 020  CPVOL1 W
```

Note: VM/370 allows cylinders normally reserved for alternate track assignment (cylinders 200 to 202 on 2319 and 2314 disks, cylinder 349 on a 3348 Data Module – Model 35, and cylinders 696 and 697 on a 3348 Data Module – Models 70 and 70F) to be optionally used for normal data storage if included within the limits of a minidisk.

Figure 15. Use and Definition of Minidisks

The real volume CPVOL1 also contains disk areas assigned to userid ABC (virtual device address 232) and userid XYZ (virtual device address 131).

| Note: On a 3330 or 3340, an OS minidisk can only start at real cylinder 0. See the list of VM/370 restrictions in the VM/370: Introduction for more information and explanation of 3330 restrictions.

MINIDISK SPACE ALLOCATION

OS bases all of its space allocation parameters on the Volume Table Of Contents (VTOC) label written on each disk; it determines the amount of space available on that volume from the format 5 (space accounting) data set control block. Thus, for OS to support minidisks, a VTOC must be written whose space accounting data set control block (DSCB) is equal to the desired size of the minidisk. The remainder of the disk space on the real disk appears to OS to be permanently dedicated, and not assignable by the OS space accounting routines. The IBCDASDI service program should be used to format minidisks for use by OS or DOS.

DOS space allocation is specified in the XTENT job control card. It is your responsibility to see that the XTENT cards refer to valid minidisk cylinders. On a 2314 or 2319 volume, the last cylinder of any minidisk initialized by IBCDASDI is always reserved for use as an alternate track cylinder. For example, if you are specifying a ten-cylinder minidisk, the XTENT card must refer to cylinders 0 through 8 only. This leaves the last cylinder for alternate track assignment.

However, on a 3330, 3333 or 3340 volume, IBCDASDI does not reserve alternate tracks. Therefore, a ten-cylinder minidisk must be defined in the XTENT card as cylinders 0 through 9.

A minidisk always begins at virtual cylinder zero. Its minimum size is one cylinder. A minidisk must exist on its real counterpart, that is, a virtual 2314/2319 minidisk must reside on a real 2314/2319.

VM/370 controls the boundaries of minidisks. If an attempt is made to refer to a DASD address outside these boundaries, CP presents a command reject (seek check) I/O error to the virtual machine.

TRACK CHARACTERISTICS

Like real disks, minidisks must be formatted for use by the appropriate service program. A minidisk is initialized for use by executing one of the following service programs in a virtual machine:

- For CMS disks, the CMS FORMAT command formats the specified tracks into 800-byte blocks or physical records.
- For CP disks, the standalone CP FORMAT program must be used to format specified tracks into 4096-byte blocks.
- For OS and DOS disks, the IBCDASDI service program writes read-only track descriptor records for each track, and clears the remaining portion of each track to binary zeroes. It also writes a Format 5 DSCB whose contents are the minidisk size.

Minidisks defined in the VM/370 directory are initialized only once; temporary minidisks must be initialized each time they are used.

ALTERNATE TRACKS

3330 DISKS

Alternate tracks flagged at the factory are automatically handled on the 3330 by the control unit. Alternate tracks on a 3330 cannot be assigned in the field. Minidisks on the 3330 Model 1 or 2 should be specified on cylinder 0 through cylinder 403 only. The remaining cylinders (404 to 411) are automatically used by the 3830 Control Unit for alternate tracks. Minidisks on the 3330 Model 11 can be specified on cylinder 0 through cylinder 807.

2314, 2319, AND 3340 DISKS

On 2314, 2319, and 3340 devices, CP and CMS do not recognize or support standard alternate track techniques for their own use. DOS and OS minidisks, however, do recognize and support alternate tracks. The IBCDASDI service program automatically assigns the last cylinder in any minidisk as an alternate track cylinder. When you initialize 2314/2319 devices, you can assign all 203 cylinders for virtual machine and system use. And for a 3340, you can initialize up to 682 cylinders, depending on the model. For a 3348 Data Module, Model 35, you can assign the entire disk, 349 cylinders. For a 3348 Data Module, Model 70, you can assign up to 682 cylinders.

If a track assigned to a virtual machine minidisk area subsequently becomes defective, you can:

- Run the standalone CP FORMAT service program and flag the whole cylinder containing the defective track as permanently assigned (PERM). This prevents CP from ever allocating that cylinder for CP paging, spooling, or temporary files. You must remember not to include this cylinder when you allocate disk space for any virtual machine's minidisk in the VM/370 directory.

- If the minidisk is used by either DOS or OS, reformat the minidisk (including the defective track) with the IBCDASDI service program. An alternate track is assigned at the end of the minidisk.

- Set up the entire pack containing the defective track as an OS or DOS pack and format it with either IBCDASDI, IEHDASDI, or IEHDASDR for OS disks, or with the DOS Initialize Disk utility program (INPP) for DOS disks. Alternate tracks are assigned in the standard manner.

LABELS

All disks to be handled by CP (as an entity or as a combination of logical disks) must have a label on real cylinder 0, track 0, record 3. This label identifies the physical volume to VM/370 and must be in the form

```
VOL1xxxxxx

 -- or --

CMS=xxxxxx
```

where xxxxxx is a six-character volume label.

In addition, all virtual machine minidisks should have a label at virtual cylinder 0, track 0, record 3. Labels created by IBCDASDI, IEHDASDR, or INPP are in the form

```
VOL1xxxxxx
```

where xxxxxx is a six-character volume label.

A physical volume that holds only virtual machine minidisks can have the first of those minidisks starting at real cylinder 0. CP recognizes the physical volume if the first minidisk has a valid label.

In Figure 15, the volume indicated as OSDOS1 has its real cylinder 0 allocated to a minidisk which is formatted for use by OS. The volume serial number of that minidisk must be OSDOS1, the label that is associated with the real volume. Since the minidisk label identifies the physical volume, changing it affects the directory entries of all users who have minidisks on that volume.

You should not assign real cylinder 0 to a user as a data area, because that user (if he has read/write access to the disk) can rewrite the label on the minidisk.

Additionally, you must not assign user minidisks to begin on real cylinder 0 of any physical volumes which are to contain CP controlled areas (for paging, spooling, and so on). On these volumes, cylinder 0 track 0 record 4 contains control information required by CP. The VTOC labels written are compatible with OS, but indicate to OS that there is no space on that DASD storage device. The initialization programs used to format OS and DOS minidisks write over and destroy this necessary control information if the space is assigned to a user minidisk, and this causes CP system failures.

SHARING MINIDISKS

A minidisk can be shared by multiple virtual machines. One virtual machine is designated the owner of the minidisk (it has an MDISK control statement in its VM/370 directory entry describing the minidisk) and other virtual machines can link to the minidisk.

For example, assume a virtual machine called USERA owns a minidisk at 194. The VM/370 directory entry for USERA contains the following statements:

```
MDISK 194 2314 050 010 SYS003 R READPASS
```

USERA's virtual disk is on the volume labeled SYS003, real cylinders 050-059.

Now, any other virtual machine that issues the LINK command or has the following LINK statement in its VM/370 directory entry can read the 194 minidisk belonging to USERA.

```
LINK USERA 194 cuu R
```

The cuu is the virtual device address at which the 194 minidisk belonging to USERA is linked to another virtual machine. If you define another virtual machine, USERB, with the following statement in its VM/370 directory entry:

```
LINK USERA 194 195 R
```

USERB reads data from USERA's 194 virtual disk whenever it issues a read for data on its own 195 virtual disk.

You can link to any minidisk that is defined in the VM/370 directory if that minidisk has a read and/or write password defined in the directory and if the type of link you desire is allowed. Three types of sharing may exist and, correspondingly, three passwords may be specified in the MDISK record.

Read-only (R) indicates that all virtual machines sharing the disk are using it in read status.

Read/write (W) indicates that multiple virtual machines may have read access at the same time.

Multi-write (M) indicates that multiple virtual machines may issue writes concurrently to the disk. Generally, this mode of access requires that the virtual machines include code to control this, such as the shared DASD support of OS. See the description of the CP LINK command in the VM/370: Command Language Guide for General Users for more information about linking to minidisks.

Configurations

Before you begin the system generation procedure, make sure your installation has the minimum configuration supported by VM/370 and the features and facilities required by VM/370.

VM/370 MINIMUM CONFIGURATION

The minimum configuration supported by VM/370 is:

One	CPU (245,760 bytes of storage)
One	System Console device
One	Printer
One	Card Reader
One	Card Punch
Two	Spindles of Direct Access Storage
One	Nine-Track Magnetic Tape Unit
One	Multiplexer Channel
One	Selector or Block Multiplexer Channel

To determine the amount of real storage and direct access storage necessary for a configuration, see "Estimating VM/370 Storage Requirements."

A representative VM/370 configuration is:

IBM 3145 512K Storage

IBM 3215 Console Printer-Keyboard

IBM 1403 Printer, Model N1 -- Two

IBM 3330 Model 1 Disk Storage or 3333 Model 1 Disk Storage and Control -- Four Drives attached to a 3830 Model 2

IBM 2305 Fixed Head Storage Facility, Model 2

IBM 3420 Magnetic Tape Units -- Two

IBM 2540 Card Read Punch

IBM 3705 Communications Controller

IBM 2741 Communication Terminals or IBM 3767 Communication Terminals, operating as 2741s (as needed)

IBM 3277 Display Stations (as needed) with the 3272 Control Unit (local attachment) or with the 3271 Control Unit (remote attachment)

IBM 3275 Display Stations (as needed for remote attachment)

Block Multiplexer Channels (#1421-1422) with Word Buffer feature (#8810) -- Two

Multiplexer Channel -- One

CONFIGURATIONS SUPPORTED BY CMS

CMS supports the following configurations:

- Virtual storage size: minimum of 320K bytes, up to 16 million bytes in multiples of 4K.
- Virtual console: any terminal supported by VM/370 as a virtual machine operator's console.
- Any virtual card readers, punches, and printers supported by VM/370 as spooling devices, except the 2520 Punch.
- Up to ten logical 2314, 2319, 3340, 3330 Model 1, 2, or 11, or 3333 Model 1 or 11 direct access storage devices. The maximum size of a CMS minidisk is:

  --246 cylinders, for a 3330 Model 1, 2, or 11

  --246 cylinders for a 3333 Model 1 or 11

  --349 cylinders for a 3348 Data Module, Model 35

  --682 cylinders for a 3348 Data Module, Model 70

  --203 cylinders (the entire disk) for a 2314/2319

- Up to four 2400, 2415, 2420, 3410 (9 track only), or 3420 (7 or 9 track) Magnetic Tape Units.

CONFIGURATIONS SUPPORTED BY RSCS

RSCS supports the following configurations:

- Virtual storage size: Minimum of 512K, up to 16 million bytes in multiples of 4K.
- Virtual console: any terminal supported by VM/370 as a virtual machine operator's console.
- Any virtual card readers, punches, and printers supported by VM/370 as spooling devices.
- One logical 2314, 2319, 3330 Model 1, 2, or 11, 3333 Model 1 or 11, or 3340 direct access storage devices.
- Transmission Control Units: 2701 Data Adapter Unit; 2703 Transmission Control Unit; or 3704 or 3705 Communications Controllers in EP mode only. Only binary synchronous communication transmission is supported.

The minimum configuration supported by RSCS is:

- 512K Virtual storage
- One console
- One Reader

- One Transmission Control Unit
- One or more binary synchronous lines dedicated to the RSCS virtual machine

DEVICES SUPPORTED BY VM/370

The following devices are supported by VM/370 except as otherwise noted. The devices are listed by device type:

- CPUs
- Direct Access Storage Devices
- Magnetic Tapes
- Unit Record Devices (Printers, Readers, and Punches)
- Terminals
- Transmission Control Units
- Remote Spooling Devices
- Other Devices

CPUS

VM/370 supports the following CPUs:

- IBM System/370 Model 135
- IBM System/370 Model 145
- IBM System/370 Model 155 II
- IBM System/370 Model 158
- IBM System/370 Model 158MP (uniprocessor mode)
- IBM System/370 Model 165 II
- IBM System/370 Model 168
- IBM System/370 Model 168MP (uniprocessor mode)

All System/370 models must have at least 245,760 bytes of real main storage.

CPU REQUIRED FEATURES AND FACILITIES

VM/370 requires the following CPU features and facilities:

- The System Timing facility (#2001), which includes the clock comparator and the CPU timer, on the Models 135 and 145.

- The Floating-point feature

  --For the Model 135, feature #3900
  --For the Model 145, feature #3910

- The Channel Indirect Data Addresssing feature on each of the 2860, 2870, and 2880 standalone I/O channels on the Model 165 II or 168.

  --For the 2860, features #1861, 1862, and 1863

  --For the 2870, feature #1861

  --For the 2880, features #1861 and 1862

  Note: The standalone channels that attach to the System/370 Models 165 II and 168 require that the Channel Indirect Data Addressing feature be ordered as a separate feature for proper operation of the input/output channels in a Dynamic Address Translation environment.

- The Word Buffer feature (#8810), available only with the System/370 Model 145, is required if

  --A 2305 Model 2 Fixed Head Storage device is attached.

  --A 3340 Direct Access Storage Facility is attached.

  --A 3330 configuration includes an Integrated File Adapter and two Selector channels, or three or more Selector channels.

DESIRABLE FEATURES

The following CPU features are desirable for VM/370:

- The Virtual Machine Assist feature improves the performance of VM/370 systems that run virtual storage operating systems in virtual

machines. It is available as a hardware feature (#8740) on the System/370 Models 135, 145, and 158, and as an RPQ (#S20573) on the System/370 Model 168.

- The Extended Floating-point feature, although not required, improves the execution of programs that use Extended Floating-point instructions under VM/370 on Models 135, 155 II, and 158.

  --For the Model 135, feature #3840
  --For the Model 155 II, feature #3700
  --For the Model 158, feature #3700

- The Word Buffer feature (#8810), available only with the System/370 Model 145, is recommended for selector channels if the 2314, 3330, or 3333 devices are attached.

DIRECT ACCESS STORAGE DEVICES

The following direct access storage devices and control units are supported by VM/370.

The direct access storage devices supported by VM/370 are:

- IBM 2305 Fixed Head Storage, Model 1 (Models 168 and 165 II only)
- IBM 2305 Fixed Head Storage, Model 2
- IBM 2314 Direct Access Storage Facility
- IBM 2319 Disk Storage
- IBM 3330 Disk Storage, Models 1, 2, and 11
- IBM 3333 Disk Storage and Control, Models 1 and 11
- IBM 3340 Direct Access Storage Facility, Models A2, B1, and B2; and the 3348 Data Modules, Models 35, 70, and 70F.

All of these direct access devices are supported as VM/370 system residence, paging and spooling devices. All except the 2305 are supported by CMS.

The following direct access control units are supported by VM/370:

- IBM 3345 Integrated Storage Control Models 3, 4, and 5 on the Model 145 for:
  - --3330 Models 1 and 2
  - --3333 Models 1 and 11
  - --3340 Model A2
- IBM 2835 Storage Control Model 1 for 2305 Model 1 (Models 165 II and 168 only)
- IBM 2835 Storage Control Model 2 for 2305 Model 2
- IBM 2844 Auxiliary Storage Control for 2314 and 2319
- IBM 3830 Storage Control Model 1 for 3330 Models 1 and 2 only
- IBM 3830 Storage Control Model 2 for 3333 Models 1 and 11, and 3340 Model A2
- IBM Integrated File Adapter (#4650) on System/370 Models 135 and 145 for 2319
- IBM Integrated File Adapter (#4655) on the System/370 Model 135 for:
  - --3330 Models 1 and 2
  - --3333 Models 1 and 11
  - --3340 Model A2
- IBM Integrated Storage Control on the System/370 Model 158 for:
  - --3330 Models 1 and 2
  - --3333 Models 1 and 11
  - --3340 Model A2

- IBM Integrated Storage Control on the System/370 Model 168:

  --3330 Models 1 and 2
  --3333 Models 1 and 11
  --3340 Model A2

- IBM 3333 Disk Storage and Control Models 1 and 11 for the 3330 Models 1, 2, and 11

- IBM 3340 Disk Storage and Control, Model A2

Notes:
1. The IBM 3330 Model 11 can be used as a system generation device in the same way as the 3330 Models 1 and 2, since the starter system does not use cylinders 404-807.
2. A System/370 Model 145 must have the Word Buffer feature (#8810) installed in order to attach a 3340 or 2305 Model 2.

MAGNETIC TAPES

The following magnetic tape devices and control units are supported by VM/370.

The magnetic tape devices supported are:

- IBM 2401, 2402, and 2403 Magnetic Tape Units
- IBM 2415 Magnetic Tape Units, Models 1, 2, 3, 4, 5, and 6
- IBM 2420 Magnetic Tape Units, Models 5 and 7
- IBM 3410/3411 Magnetic Tape Unit, Models 1, 2, and 3 (9-track only)
- IBM 3411 Magnetic Tape Unit and Control, Models 1, 2, and 3 (9-track only)
- IBM 3420 Magnetic Tape Units, Models 3, 4, 5, 6, 7, and 8

The magnetic tape control units supported are:

- IBM 2803 Tape Control
- IBM 2804 Tape Control
- IBM 3411 Magnetic Tape Unit and Control
- IBM 3803 Tape Control

UNIT RECORD DEVICES

VM/370 supports the following printers, readers, punches, and unit record control units.

VM/370 supports the following printers:

- IBM 1403 Printer Models 2, 3, 7, and N1 (with minimum of 132 print positions)
- IBM 1443 Printer Model N1 (with 144 print positions)
- IBM 3211 Printer (Right Indexing only)
- IBM 3213 Printer (in 3215 Emulator Mode)

VM/370 supports the following readers/punches:

- IBM 2501 Card Reader Models B1 and B2
- IBM 2520 Punch Models B2 and B3
- IBM 2540 Card Read Punch Model 1
- IBM 3505 Card Reader Models B1 and B2
- IBM 3525 Card Punch Models P1, P2, and P3

VM/370 supports the following unit record control units:

- IBM 2821 Control Unit
- IBM 3811 Printer Control Unit
- IBM Integrated Printer Adapter (IPA) (#4662, 4667, or 4668) on the System/370 Model 135

TERMINALS

The following system consoles are supported by VM/370 as virtual system consoles (simulated as 3215 consoles).

- IBM 2150 Console with 1052 Printer-Keyboard Model 7
- IBM 3066 System Consoles Models 1 and 2 for the System/370 Models 165 II and 168
- IBM 3210 Console Printer-Keyboard Models 1 and 2
- IBM 3215 Console Printer-Keyboard Model 1
- IBM System Console for the System/370 Model 158 (in printer-keyboard mode with the 3213 Printer Model 1 required, or in display mode)
- IBM 7412 Console (via RPQ AA2846) with 3215 Console Printer-Keyboard Model 1

Note: During system generation only, the primary system operator's console cannot be connected to the system via a teleprocessing line.

The following terminals are supported by VM/370 as virtual system consoles (simulated as 3215 consoles):

- IBM 2741 Communication Terminal
- IBM 1050 Data Communication System
- Terminals on switched lines compatible with the line control used by the IBM Telegraph Control Type II Adapter (8-level ASCII code at 110 bps) such as the CPT-TWX (Model 33/35) terminals
- IBM 3275 Display Station, Model 2 with integral control unit (remote attachment only)
- IBM 3277 Display Station, Model 2, via 3272 Control Unit, Model 2 (local attachment only)
- IBM 3277 Display Station, Model 2, via 3271 Control Unit, Model 2 (remote attachment only)
- IBM 3767 Communications Terminal, Model 1 (operating as a 2741)

SPECIAL CONSIDERATIONS AND REQUIRED FEATURES

Terminals that are equivalent to those explicitly supported may also function satisfactorily. You are responsible for establishing equivalency. IBM assumes no responsibility for the impact that any changes to the IBM-supplied programs or products may have on such terminals.

Prior availability of an RPQ does not guarantee or imply current or future availability. Contact your IBM branch office for ordering information concerning the RPQs mentioned with the following features.

Configurations (Terminals)

2741 Features: Supported, Required, and Desirable Features

The IBM 2741 Communication Terminal is supported on either a switched or point-to-point nonswitched lines with these features:

- PTTC/EBCD (#9571, Part #1167963) or standard Correspondence (#9812, Part #1167043) print elements
- Transmit Interrupt (#7900) or Transmit Interrupt Control RPQ #E40681
- Receive Interrupt (#4708)
- For switched lines, the Data Set Attachment (#9114) and Dialup feature (#3255) are required
- For point-to-point nonswitched lines, one of the following features is required:

  --Data Set Attachment (#9115 for facility D1), or
  --Data Set Attachment (#9116 for facility B2), or
  --Data Set Attachment (#9120 for facility B1 or D1), or
  --IBM Line Adapter (#4635 for 4-wire limited distance line), or
  --IBM Line Adapter (#4691-4694 for 4-wire shared nonswitched line)

  -- or --

  --IBM Line Adapter (#4647 for 4-wire nonswitched line)

The following features, although not required, enhance the convenience and usability of the terminal:

- Print Inhibit (#5501)
- Red Ribbon Control RPQ #868019 (supported for PTTC/EBCD keyboard only)
- Typamatic Keys (#8341)
- Pin Feed Platen (#9509)

1050 Control Units, Models, and Features: Supported, Required, and Desirable Features

The IBM 1050 Data Communication System is supported on either switched or point-to-point nonswitched lines with these features:

- IBM 1051 Control Unit (Model 1 or 2) with these features:

  --Transmit Interrupt (#7900) or Transmit Interrupt Control RPQ #E26903

  --Receive Interrupt (#6100) or Receive Interrupt Control RPQ #27428

  --Text Time-Out Suppression (#9698)

  --First Printer Attachment (#4408). This feature is required to attach a 1052 Printer-Keyboard to the 1051

- IBM 1052 Printer-Keyboard (Model 1 or 2) with the PTTC/EBCD print element (#9571, Part #1167963)
- For switched lines, the Data Set Attachment (#9114) is required

- For point-to-point nonswitched lines, one of the following is required:

  --Data Set Attachment (#9115 for facility D1)

  --or--

  --Data Set Attachment (#9116 for facility B2)

  --or--

  --Data Set Attachment (#9120 for facility B1 or D1)

  --or--

  --IBM Line Adapter (#4691-4694 for 4-wire shared nonswitched line)

  --or--

  --IBM Line Adapter (#4647 for 4-wire nonswitched line)

  The following features, although not required, enhance the convenience and usability of the terminal:

- Automatic Ribbon Shift and Line Feed Select (#1295)
- Automatic EOB on Carrier Return RPQ #E 28235

3270 Components, Control Units, Models, and Features: Supported, Required, and Desirable Features

The IBM 3270 Information Display System is supported on a byte multiplexer, block multiplexer or selector channel with these components:

- IBM 3271 Control Unit, Model 2, for remote attachment of 3277 display stations

- IBM 3272 Control Unit, Model 2, for local attachment of 3277 display stations

- IBM 3275 Display Station, Model 2 (remote attachment), or
  IBM 3277 Display Station, Model 2 (local or remote attachment), with one of the following features required:

  --66 Key EBCDIC Typewriter Keyboard (#4630)
  --66 Key EBCDIC Data Entry Keyboard (#4631)
  --78 Key Operator Console - Keyboard (#4632)
  --78 Key EBCDIC Typewriter Keyboard (#4633)

  The following features, while not required, enhance the convenience and usability of this terminal:

  --Keyboard Numeric Lock (#4690)
  --Audible Alarm (#1090)
  --Operator Identification Card Reader (#4600)
  --Lower Case Character Display (RPQ #8K0366)

  Note: Unless a 3270 terminal is dedicated to a virtual machine, it is supported only as a virtual 3215; that is, it is not supported by VM/370 as a real local or remote 3270.

- IBM 3284 Printer, Models 2 and 3 (for remotely attached 3270s)
- IBM 3286 Printer, Models 2 and 3 (for remotely attached 3270s). The Model 3 attaches to the 3275 display station only if RPQ MB4317 is installed.
- IBM 3288 Line Printer, Model 2 (for remotely attached 3270s)

3767 Features: Required and Desirable Features

The IBM 3767 Communication Terminal, Model 1, is supported when it operates as an IBM 2741 Communication Terminal and is attached to a 3704 or 3705 Communications Controller. It requires the following features on either switched or nonswitched point-to-point lines:

- 2741 START/STOP (#7113)
- EBCDIC (#9391) or Correspondence (#9381) keyboard
- Half-duplexed, switched or nonswitched line (#9402)
- One of the following:

  --EIA Interface with Clock (#3719) at 300 bps
  --1200 bps Integrated Modem/Interrupt (#5505 or #5500)

The following features, although not required, enhance the convenience and usability of the terminal:

- Alternate Character Set (#1291), plus a defined character subset for the keyboard:

  --If the primary character set is Correspondence (#9381), the alternate character set can be APL (#9383) or EBCDIC (#9382).

  --If the primary character set is EBCDIC (#9391), the alternate character set can be APL (#9393) or Correspondence (#9392).

  Note: Line control is PTTC/EBCD with this feature.

- Acoustic Coupler (#1110) at 300 bps #9540

TRANSMISSION CONTROL UNITS

VM/370 supports the following transmission control units:

- IBM 2701 Data Adapter Unit
- IBM 2702 Transmission Control
- IBM 2703 Transmission Control
- IBM Integrated Communications Attachment (ICA), (#4640)
- IBM 3704 and 3705 Communications Controllers

2701 REQUIRED FEATURES

- For line control of CPT-TWX (Model 33/35) terminals, the Telegraph Adapter Type II (#7885) is required.
- For 2770, 2780, 3770 (as a 2770), and 3780 terminals, the following are required:

  --Synchronous Data Adapter Type II (#7698)

  --EBCDIC code (#9060)

  --EBCDIC transparency (#8029)

- For 1050 and 2741 terminals, the following are required:

  --IBM Terminal Adapter Type I, Model II (#4640)

  --Selective Speed, 134.5 bps (#9581)

  --2741 Break Feature RPQ #M53193, and Break Command RPQ #858492

- The Expanded Capability feature (#3815) is required if there are:

  --More than two low speed adapters (either IBM Type I Model II, or Telegraph Type II), or

  --More than one high speed adapter (Synchronous Data Adapter Type II), or

  --One high speed and at least one low speed adapter attached to the same 2701.

- The Expansion Feature (#3855) is required for each line adapter after the first.

2702 REQUIRED AND OPTIONAL FEATURES

- For 1050 and 2741 terminals, the following are required:

  --Terminal Control Base for IBM Terminal Control (#9696)

  --IBM Terminal Control Type I (#4615)

  --Selective Speed, 134.5 bps (#9684)

  --Type I Terminal Interrupt (#8200)

Configurations (TCUs)

--Data Set Line Adapter (#3233) or IBM Line Adapter (#4635), 4-wire IBM Terminal Control Type I (#4615)

- For line control of CPT-TWX (Model 33/35) terminals, the following are required:

  --Terminal Control Base for Telegraph Terminal Control (#9697)

  --Telegraph Terminal Control Type II (#7912)

  --Pluggable End Characters (return key generates an interrupt) RPQ #E62920, optional

  --Data Set Line Adapter (#3233)

  --Terminal Control Expansion (7935), required only if both of the terminal bases (#9697 and #7912) are attached to the same 2702.

- The 31 Line Expansion (#7955) is supported as needed.

2703 REQUIRED AND OPTIONAL FEATURES

- For 1050 and 2741 terminals, the following are required:

  --Start-Stop Base Type I (#7505) or Type II (#7506)

  --IBM Terminal Control Base (#4619)

  --IBM Terminal Control Type I (#4696)

  --Line Speed Option, 134.5 bps (#4878)

  --Type I Terminal Interrupt (#8200)

  --Data Line Set (#3205) and/or IBM Line Set 1B (#4687)

- For line control of CPT-TWX (Model 33/35) terminals, the following are required:

  --Telegraph Terminal Control Base (#7905)

  --Telegraph Terminal Control Type II (#7912)

  --Line Speed Option, 110 bps (#4877)

  --Data Line Set (#3205), and Data Line Set Expander (#3206)

  --Pluggable End Characters (return key generates an interrupt) RPQ #E66707, optional

- For 2770, 2780, and 3780 Terminals, the following are required:

  --Synchronous Base (#7703, 7704, or 7706)

  --Synchronous Terminal Control for EBCDIC (#7715)

  --Transparency (#9100)

  --Synchronous Line Set (#7710)

- The Base Expansion feature (#1440) is required if more than one base type is to be attached to the same 2703.

IBM INTEGRATED COMMUNICATIONS ATTACHMENT (ICA) REQUIRED AND OPTIONAL FEATURES

The ICA (#4640) is available on the System/370 Model 135. Additional lines (#4722-4728) are supported.

- For 1050, 2741, and 3767 (as a 2741) terminals, the following are required:

  --Terminal Adapter Type I Model II (#9721-9728)

  --Switched Network Facility (#9625-9632), optional

  --Write Interrupt (#9745-9752)

  --Read Interrupt (#9737-9744)

  --Unit Exception Suppression (#9729-9730), optional

  --For the 3767 only, as a 2741, 200 bps (#2711-2718) or 300 bps (#9593-9600)

- For 2770, 2780, 3770 (as a 2770) and 3780 terminals, the following are required:

  --Synchronous Data Adapter Type II (#9649-9656)

  --Half Duplex Facility (#9617-9624)

  --EBCDIC Transparency (#9673-9680)

- For line control of CPT-TWX (Model 33/35) terminals, the following are required:

  --Telegraph Adapter Type II (#9785-9792)

  --Switched Network Facility (#9625-9632)

3704/3705 REQUIRED FEATURES

The IBM 3704 and 3705 Communications Controllers are supported in Network Control Program mode, Partitioned Emulation Program mode, and 2701, 2702, 2703 Emulation Program mode.

Note: VM/370 supports the CPT-TWX (Model 33/35) terminals only at 110 bps when when attached to a 3704 or 3705.

VM/370 supports all models of the 3704 and 3705 Communications Controllers. However, because the network control program and partitioned emulation program require 48K of storage, the following models are supported for the emulation program only.

- IBM 3704 Communications Controller Models A1 and A2
- IBM 3705 Communications Controller Models A1, B1, C1, and D1

The features required on a communications controller do not depend on VM/370. VM/370, when supporting network control mode, requires a communications controller with at least 48K of storage. Other 3704/3705 features depend upon the intended use of the communications controller and the type of 3704/3705 control program (emulation, network control or partitioned emulation program) to be executed.

VM/370 does not support the following 3704/3705 features:

- Line Set Type 2A (#4721)
- Line Set Type 3A (#4731)
- Line Set Type 4B (#4742)

REMOTE SPOOLING DEVICES SUPPORTED BY VM/370

Remote spooling is supported in VM/370 by the 2780 standalone program (DMKSRP) and the Remote Spooling Communications Subsystem (RSCS).

2780 SPOOL REMOTE PROGRAM (DMKSRP)

VM/370 supports the IBM 2780 Data Transmission Terminal Model 2, for standalone remote spooling (the DMKSRP Spool Remote Program), with these features:

- EBCDIC Transmission Code (#9762)
- EBCDIC Transparency (#8030)
- 144 Character Print Line (#5820, 5821)
- Automatic Turnaround (#1350), if the 2780 is using a switched line facility

REMOTE SPOOLING COMMUNICATIONS SUBSYSTEM (RSCS)

The VM/370 Remote Spooling Communications Subsystem (RSCS) supports the following:

- IBM 2770 Data Communication Terminal
- IBM 2780 Data Transmission Terminal, Models 1 and 2
- IBM 3770 Data Communication System (as a 2770)
- IBM 3780 Data Communication Terminal
- HASP supported programmable workstations

2770 Features: Required and Optional Features

The IBM 2770 Data Communication System with the 2772 Multipurpose Control Unit can be connected to the central System/370 via a switched or nonswitched point-to-point communication line.

The following devices and features are required for operating a 2770 as an NPT remote station:

- One IBM 2213 Printer, Model 2, or one IBM 2203 Printer, or one IBM 1503 Printer
- One IBM 2502 Card Reader, Model A1 or A2
- EBCDIC Transmission Code (#9761)

Other supported equipment and features are:

- One IBM 545 Card Punch, Model 3 or 4, with or without 3590 attachment
- EBCDIC Transparency (#3650)
- Additional Buffer Expansion (#1491)
- Space Compression/Expansion (#6555)
- Synchronous Clock (#7705)

2780 Features: Required and Optional Features

The IBM 2780 Data Transmission Terminal, Models 1 and 2, can be connected to the central System/370 via a switched or nonswitched point-to-point line. EBCDIC Transmission code (#9762) is required.

The following features are optional:

- EBCDIC Transparency (#8030)
- 120/144 Character Print Line (#5820 or #5821)
- Multiple Record Transmission (#5010)
- Synchronous Clock (#7705)

3770 Features

The IBM 3770 Data Communication System can be connected to the central System/370 via a switched or nonswitched point-to-point communications line. Required features are:

- EBCDIC Transmission Code (#9761)
- SDLC/BSC Switch Control (#1460)
- BSC point-to-point (#1461)

3780 Features: Required and Optional Features

The IBM 3780 Data Communications Terminal can be connected to the central System/370 via a switched or nonswitched point-to-point communications line. EBCDIC transmission code (#9761) is required.

The following devices and features are optional:

- One IBM 3781 Card Punch
- Component Selection (#1601, required for the 3781)
- EBCDIC Transparency (#3601)
- Additional Print Positions (#5701)
- Synchronous Clock (#7705)

Programmable Terminals

RSCS Spool MULTI-LEAVING[1] (SML) supports, as a VM/370 remote workstation, any processor that is supported as a HASP workstation and is programmed to operate as a HASP workstation.

---

[1]IBM Unregistered Trademark

CPUs Supported

- The IBM 1130 Computing System (except Models 4A and 4B) is supported if it has at least 8K words of storage and the Synchronous Communications Adapter (#7690).
- Any System/360 or System/370 is supported if it has at least 8K bytes of main storage and a 2701, 2703 or 3704/3705 in EP mode, equipped for EBCDIC transmission and binary synchronous communications.
- Any submodel of the System/360 Model 20 or the IBM 2922 is supported if it has at least 8K bytes of main storage and the following features:

  --Binary Synchronous Communications Adapter (#2074)

  --EBCDIC Transmission code (#9060)

  --Full Transparency (#4100)
- IBM System/3 is supported with the following features:

  --Binary Synchronous Communications Adapter (#2074)

  --EBCDIC Transmission code (#9060)

  --Text transparency (#7850)

OTHER CONSIDERATIONS FOR PLANNING YOUR CONFIGURATION

TWO-CHANNEL SWITCH

VM/370 has no multiple path support and does not take advantage of the two-channel switch. However, a two-channel switch can be used between the System/370 running a virtual machine under VM/370 and another CPU.

If any I/O devices controlled by VM/370 for its exclusive use are attached to control units with two-channel switches, the CPU or virtual machine controlling the other channel interface must vary the CP-owned devices offline.

See the "VM/370 Using Channel Switching" section earlier in Part 1 for more information about using the two-channel switch.

DEVICES USED ONLY BY AN OPERATING SYSTEM IN A VIRTUAL MACHINE AND NOT BY VM/370

Any input/output device that can be attached to the IBM System/370 can be used by a virtual machine under VM/370 as long as there are:

- No timing dependencies in the device or the program
- No dynamically modified channel programs except OS Indexed Sequential Access Method (ISAM) or OS/VS Telecommunications Access Method (TCAM) Level 5
- None of the other restrictions outlined in the VM/370: Introduction are violated.

Dynamically modified channel programs (except those that have input/output involving page zero) are permitted if run in a virtual=real machine.

Input/output devices that are part of a virtual machine's configuration require real device equivalents, except for:

- Unit record devices, which CP can simulate using spooling techniques.
- Virtual 2311 Disk Storage Drives, which CP can map onto 2314 or 2319 disks. Up to two full 2311 units can be mapped onto a 2314 or 2319 disk in this manner.

Part 2: Defining Your VM/370 System

Part 2 describes the macros and control statements you need to define your VM/370 system. It contains the following sections:

- Preparing the Real I/O Configuration File (DMKRIO)
- Preparing the CP Control File (DMKSYS)
- Creating Your VM/370 Directory
- Preparing the System Name Table File (DMKSNT)
- Altering the Forms Control Buffer Load (DMKFCB)

Before starting the system generation procedures on a real machine, you must create three files that describe the VM/370 system you are generating. There are two additional files, which are optional. You can use card input, or create these files using the CMS Editor. If you are modifying an existing VM/370 system, you can use the CMS Editor to alter the existing files.

The three files that you must prepare are:

- The real I/O configuration file (module name DMKRIO), which defines the I/O configuration on the real System/370 machine.
- The CP system control file (module name DMKSYS), which defines CP-controlled DASD volumes, allocation, and so on.
- The VM/370 directory file (normally a CMS file named USER DIRECT), which contains the VM/370 directory entries that define the virtual machine configuration for each user.

In addition, you should prepare the system name table (DMKSNT) file if you plan to save systems. If you generate the 3704/3705 control program, you must save it; otherwise, the 3704/3705 control program cannot be loaded by VM/370. "Part 1: Planning for System Generation" has a section that describes the requirements for saving systems.

You can also change the forms control buffer load (module name DMKFCB).

The notational conventions that describe the macro syntax for VM/370 system generation are listed in "Appendix E. Notational Conventions." These notational conventions are the same as the conventions used to describe VM/370 commands.

All the groups of CLUSTER and TERMINAL macros must appear first, followed by all RDEVICE macros, all RCTLUNIT macros, all RCHANNEL macros, and finally by the RIOGEN macro. In addition, the first statement in the file must be the DMKRIO CSECT statement (as shown) and the last statement must be the assembler END statement.

CODING THE REAL I/O CONFIGURATION MACROS FOR REMOTE 3270S

Two types of remote 3270 configurations are supported: a clustered 3271 station with multiple terminals and printers attached and a standalone 3275 station. These remote 3270 stations are attached via binary sysnchronous communications lines.

To define remote 3270 stations you must code CLUSTER, TERMINAL, and RDEVICE macros. Code one RDEVICE macro for each binary synchronous line that supports a remote 3270 configuration. Code one CLUSTER macro to define the 3270 control unit for each of those lines and code one or more TERMINAL macros, as needed, to define the devices in the remote 3270 configuration.

Code one CLUSTER macro for each binary synchronous line used by remote 3270s. The CLUSTER macro defines the control unit (3271 or 3275) for the remote 3270 configuration. Each CLUSTER macro must have a unique label. This label is coded on the RDEVICE macro that defines the corresponding binary synchronous line and thus logically links the line and the cluster. The address of the line (defined by the ADDRESS=cuu operand of the RDEVICE macro) is coded in the LINE=cuu operand of the CLUSTER macro.

Follow each CLUSTER macro with the TERMINAL macros that define the terminals for the remote 3270 control unit. For the 3271, directly following the CLUSTER macro, code a TERMINAL macro for each terminal address to which a terminal can be attached (regardless of whether or not the intermediate addresses are unused). For example, if terminals are attached to the third, fourth, eighth, and ninth addresses, you code nine TERMINAL macros. The first macro represents the first (lowest) address, the last represents the ninth (highest) address.

For the 3275, directly following the CLUSTER macro, code a single TERMINAL macro specifying TERM=3275. If the 3275 has a 3284 or 3286 Model 3 Printer attached, specify MODEL=3 to define the printer; otherwise, the printer is ignored.

After all the CLUSTER-TERMINAL groups of macros have been coded, code the other real I/O configuration macros. You must code an RDEVICE macro for each binary synchronous line that supports remote 3270 stations. Specify the label of the corresponding CLUSTER macro on the RDEVICE macro (CLUSTER=label).

CLUSTER MACRO

Use the CLUSTER macro to define remote 3270s. Each CLUSTER macro represents a clustered (3271) or standalone (3275) station on a nonswitched point-to-point binary synchronous communication line. Only one CLUSTER macro can be specified for each communication line.

The format of the CLUSTER macro is:

Name	Operation	Operands
label	CLUSTER	CUTYPE={3271 / 3275}
		,GPOLL=cudv
		,LINE=cuu

where:

label
is a name of the CLUSTER macro; it must be specified. The label may be any valid assembler language symbol. The label establishes a unique symbolic name for this cluster or standalone station.

CUTYPE={3271 / 3275}
is the control unit of the station; it is either 3271 or 3275.

GPOLL=cudv
are the general polling characters that represent the general polling technique to be used for this station. When general polling is used, the first device that the control unit determines is ready to send data over the line is allowed to do so. The characters, cudv, are the four-digit hexadecimal general polling characters assigned to the control unit of the station. The hexadecimal equivalent of the EBCDIC transmission code in in the form cudv, where:

cu are the polling characters for the control unit.
dv are the characters for any available input device (normally X'7F')

The general polling characters for the remote 3270 device (dv) are always X'7F' and the general polling characters for the control unit are defined when the control unit is physically installed. Use Figure 15.1 to determine what you should code as the general polling characters for the control unit.

LINE=cuu
is the line interface address. It is the address specified on the RDEVICE macro that is associated with this CLUSTER macro.

Examples

The following CLUSTER macro describes a 3271 control unit with a control unit address of 2 and a line address of 078.

```
CLUST001 CLUSTER CUTYPE=3271,GPOLL=C27F,LINE=078
```

The following CLUSTER macro describes a 3275 display station with a control unit address of 0 and a line address of 080.

```
CLUST020 CLUSTER CUTYPE=3275,GPOLL=407F,LINE=080
```

In the real I/O configuration file (DMKRIO), the CLUSTER macro must immediately precede the TERMINAL macros that define the stations attached to that cluster or standalone station.

MNOTE Messages

The following MNOTE messages may be generated by the CLUSTER macro instructions.

INVALID CUTYPE OPERAND
: indicates that the control unit is not specified correctly: 3271 and 3275 are the only valid operands.

CUTYPE NOT SPECIFIED
: indicates that the required operand, CUTYPE=3271 or 3275, is missing.

INVALID GPOLL OPERAND
: indicates that the general polling characters are specified incorrectly. See Figure 15.1 for the valid general polling characters.

GPOLL NOT SPECIFIED
: indicates that the required operand, GPOLL=cudv, is missing.

INVALID LINE OPERAND
: indicates that one or more of the line addresses is specified incorrectly.

LINE NOT SPECIFIED
: indicates that the required operand, LINE=cuu, is missing.

NAME FIELD NOT SPECIFIED
: indicates that the label field is not coded on one or more CLUSTER macro. The name field, or label, must be specified on all CLUSTER macros.

MORE THAN 16 LINES FOR REMOTE CLUSTERS
: indicates that more than 16 CLUSTER macros are in the DMKRIO file: 16 is the maximum allowed.

CLUSTER MACRO OUT OF SEQUENCE
: indicates that a CLUSTER macro is out of sequence. Each CLUSTER macro must immediately precede the TERMINAL macros defining the devices attached at each remote 3270 station. The groups of CLUSTER and TERMINAL macros must precede all other macros in the DMKRIO file.

TERMINAL MACRO

Use the TERMINAL macro to define (1) a display station or printer that is attached to the remote 3270 display system or (2) a terminal address that is available to attach an additional remote 3270. Each terminal address attached to a cluster must be represented by a TERMINAL macro. Only one TERMINAL macro is specified for a standalone display station.

Code one TERMINAL macro for each display device and each printer attached to a clustered control unit (3271). You must code a TERMINAL macro for every terminal address to which a terminal can be attached, even if a terminal address is unused. When you code a TERMINAL macro for an unused terminal address, specify a valid TERM= operand and the correct selection or addressing characters.

Code only one TERMINAL macro to define the display station, and optionally a printer, attached to a standalone station (3275). Code TERM=3275 to define the 3275 display station and, optionally, code MODEL=3 to define a 3284 or 3286 printer attached to the 3275.

The format of the TERMINAL macro is:

Name	Operation	Operands
	TERMINAL	TERM={3275 / 3277 / 3284 / 3286 / 3288}
		,SELECT=cudv
		[,MODEL=2 / ,MODEL=3]
		[,FEATURE=OPRDR]

where:

TERM={3275 / 3277 / 3284 / 3286 / 3288}
is the device type of the remote 3270 station attached to the clustered or standalone 3270 control unit. The following devices are allowed:

Device	TERM=
IBM 3275 Display Station	3275
IBM 3277 Display Station	3277
IBM 3284 Printer	3284
IBM 3286 Printer	3286
IBM 3288 Line Printer	3288

SELECT=cudv
are the four-digit hexadecimal selection or addressing characters assigned to this device, where:

cu are the characters for the control unit
dv are the characters for the device

Use Figure 15.1 to determine the selection and addressing characters for this device.

Note: If a printer is attached to the 3275, it has the same address as the 3275 display station.

[MODEL=2]
[MODEL=3]
is the model number of the printer; the default is model 2.

The following printers can be attached to a 3271 clustered control unit:

- IBM 3284 Printer, Model 2
- IBM 3286 Printer, Model 2
- IBM 3288 Printer, Model 2

The following printers can be attached to a standalone 3275 station:

- IBM 3284 Printer, Model 3
- IBM 3286 Printer, Model 3 (via RPQ MB4317)

FEATURE=OPRDR
specifies the optional feature, operator identification card reader, that is available on the 3277 Display Station, Model 2.

Examples

Example 1: This TERMINAL macro describes a 3277 with a selection address of 2, and a control unit address of 2.

```
TERMINAL TERM=3277,SELECT=E2C2,FEATURE=OPRDR
```

Example 2: This TERMINAL macro describes a 3286 with a selection address of 3 and a control unit address of 3.

```
TERMINAL TERM=3286,SELECT=E3C3
```

Example 3: This TERMINAL macro describes a 3284 with a selection address of 4 and a control unit address of 4.

```
TERMINAL TERM=3284,SELECT=E4C4,MODEL=2
```

Example 4: This TERMINAL macro describes a 3275 Display Station with a 3284 Printer, Model 3, attached and a control unit address of 0.

```
TERMINAL TERM=3275,SELECT=6040,MODEL=3
```

If no printer is attached to the 3275, code:

```
TERMINAL TERM=3275,SELECT=6040
```

MNOTE Messages

The following MNOTE messages may be generated by the TERMINAL macro instructions.

INVALID 'TERM' OPERAND
: indicates that the device type is not specified correctly; 3275, 3277, 3284, 3286, and 3288 are the only valid operands.

'TERM' NOT SPECIFIED
: indicates that the required operand, TERM=32xx, is missing.

INVALID 'SELECT' OPERAND
: indicates that the selection or addressing characters are specified incorrectly. See Figure 15.1 for the valid selection or addressing characters.

'SELECT' NOT SPECIFIED
: indicates that the required operand, SELECT=cudv, is missing.

INVALID 'MODEL' NUMBER
: indicates that the model number is specified incorrectly; 2 and 3 are the only valid model numbers.

INVALID 'FEATURE' OPERAND
: indicates that the feature operand is specified incorrectly; OPRDR is the only valid feature.

TERMINAL MACRO OUT OF SEQUENCE
: indicates that a TERMINAL macro is out of sequence. All the TERMINAL macros defining the devices attached to a remote 3270 station must follow the CLUSTER macro that defines the control unit for that station. The groups of CLUSTER and TERMINAL macros must precede all other macros in the DMKRIO file.

Use this column for: • Device selection • Specific poll • General poll • Fixed return addresses		Use this column for: • 3270 control unit selection addresses	
If the Control Unit or Device Number is:	The EBCDIC Code (in hexadecimal) is: See Note 1	If the Control Unit Number is:	The EBCDIC Code (in hexadecimal) is: See Note 1
0	40	0	60
1	C1	1	61
2	C2	2	E2
3	C3	3	E3
4	C4	4	E4
5	C5	5	E5
6	C6	6	E6
7	C7	7	E7
8	C8	8	E8
9	C9	9	E9
10	4A	10	6A
11	4B	11	6B
12	4C	12	6C
13	4D	13	6D
14	4E	14	6E
15	4F	15	6F
16	50	16	F0
17	D1	17	F1
18	D2	18	F2
19	D3	19	F3
20	D4	20	F4
21	D5	21	F5
22	D6	22	F6
23	D7	23	F7
24	D8	24	F8
25	D9	25	F9
26	5A	26	7A
27	5B	27	7B
28	5C	28	7C
29	5D	29	7D
30	5E	30	7E
31	5F	31	7F

Note 1: Graphic characters for the United States I/O interface codes are shown. Graphic characters for EBCDIC 4A, 5A, 5B, 7B, 7C, and 7F might differ for particular World Trade I/O interface codes.

Figure 15.1. Remote 3270 Control Unit and Device Addressing

Figure 15.2 shows some examples of valid polling characters.

3271 Addressing			3275 Addressing		
General Poll for Control Unit 5	Control Unit Address	EBCDIC C5 C5	General Poll for Control Unit 5	Control Unit Address	EBCDIC C5 C5
	Device Address	7F 7F		Device Address	7F 7F
Specific Poll Device 4 on Control Unit 5	Control Unit Address	C5 C5	Specific Poll for Control Unit 5	Control Unit Address	C5 C5
	Device Address	C4 C4		Device Address	40 40
Select Device 4 on Control Unit 5	Control Unit Address	E5 E5	Select Control Unit 5	Control Unit Address	E5 E5
	Device Address	C4 C4		Device Address	40 40

Figure 15.2. Examples of Remote 3270 Addressing

Preparing the Real I/O Configuration File (DMKRIO)

The real I/O configuration file consists of macros that describe the I/O devices, control units, and channels attached to the real System/370. VM/370 uses this information to schedule I/O operations and to allocate resources. Therefore, the real I/O macro entries must represent the real hardware configuration accurately. Generally, there must be one real I/O macro entry for each hardware unit in your configuration.

You can include entries for more devices than your installation has so that devices can be added in the future without performing another system generation, but bear in mind that the control blocks generated (RDEVBLOK, RCUBLOK, and RCHBLOK) occupy space in real storage.

When preparing the RDEVICE and RCTLUNIT entries, refer to "Appendix B: Configuration Aid" to assist you in configuring contrcl units and devices. Following the descriptions of the CLUSTER, TERMINAL, RDEVICE, RCTLUNIT, RCHANNEL, and RIOGEN macros, there is an example showing how these macros are coded for one particular real configuration.

The macros, in their proper sequence, are:

Macro Name	Units Referred To
CLUSTER	Remote Display Stations
TERMINAL	
.	
.	
.	
RDEVICE	I/O Devices
RCTLUNIT	Control Units
RCHANNEL	Channels
RIOGEN	System Console

The file is placed in the reader as follows:

```
DMKRIO CSECT
       CLUSTER macro[1]
       TERMINAL macro[1]
          .
          .
          .
       RDEVICE macros
          .
          .
          .
       RCTLUNIT macros
          .
          .
          .
       RCHANNEL macros
          .
          .
          .
       RIOGEN macro
       END
```

---

[1]There must be a CLUSTER macro for each binary synchronous line for remote 3270s. Each CLUSTER macro must be followed immediately by the TERMINAL macros representing each display station and printer on that line. The CLUSTER and TERMINAL macro groups must precede all the other real I/O configuration macros.

RDEVICE MACRO

Use the RDEVICE macro instruction to generate a real device block (RDEVBLOK). You must code an RDEVICE macro for each real I/O device in your I/O configuration. The maximum number of real devices that can be included on the real VM/370 system is 3276.

The RDEVICE macro instructions describe each device, or group of devices, attached to the System/370. These can be in any order, but they must be contiguous, and must precede all RCTLUNIT and RCHANNEL macros in the real I/O configuration file (DMKRIO). Also, the RDEVICE macro instructions must follow all the groups of CLUSTER and TERMINAL macros, if there are any. The first RDEVICE macro generates the label DMKRIODV, which indicates the start of the real device blocks to CP.

The name field may not be specified for the RDEVICE macro instruction; if a name is specified it is ignored. The RDEVICE macro generates a name by appending the device address to the characters RDV. For example, the name RDV234 is generated for the device address 234.

Before you code an RDEVICE macro for a 3704 or 3705 device, see the "Generating a VM/370 System that Supports the 3704/3705" section of Part 1 for additional information and special considerations.

Also, see the "Planning Considerations for Remote 3270s" section of Part 1 before you code an RDEVICE macro for a binary synchronous line that is used by remote 3270s.

The format of the RDEVICE macro is:

```
Name | Operation | Operands
     |  RDEVICE  | ADDRESS={cuu      },DEVTYPE=type[,MODEL=model]
     |           |         {(cuu,nn) }
     |           | [,FEATURE=(feature[,feature]...)]
     |           | [,CLASS={(cl[,cl]...)}]
     |           |         {DASD       }
     |           |         {TAPE       }
     |           |         {TERM       }
     |           |         {GRAF       }
     |           |         {URI        }
     |           |         {URO        }
     |           | [,ADAPTER={BSCA }]
     |           |           {IBM1 }
     |           |           {TELE2}
     |           |           {TYPE1}
     |           |           {TYPE2}
     |           | [,SETADDR=sadnum]
     |           | [,CPTYPE={EP }]
     |           |          {NCP}
     |           |          {PEP}
     |           | [,CPNAME=cpname]
     |           | [,MAXDIAL=nn]
     |           | [,BASEADD=cuu]
     |           | [,CLUSTER=label]
```

where:

ADDRESS={cuu | (cuu,nn)}
is the real I/O device address or (addresses).

The address, cuu, is three hexadecimal digits from 000 to FFF. The high order digit is the address of the channel to which the device is attached. The low order two digits represent the control unit and device address.

The value, nn, is the number of RDEVBLOK entries to be generated; it may be any number from 1 to 256. For example, if ADDRESS=(10,5) is specified, RDEVBLOKs with device address 10, 11, 12, 13, and 14 are generated. If nn is omitted, a value of 1 is assumed for all devices except the 2314 and 2305, which have a default value of 8.

If DEVTYPE=3066 or 3158, nn can only be 1 because only one 3066 or only one 3158 can be specified for each RDEVICE macro.

DEVTYPE=type
is the type of device.

The device type can be ICA, CTCA, 1017, 1018, 1052, 1053, 1403, 1442P, 1442R, 1443, 2150, 2250, 2260, 2265, 2301, 2303, 2305, 2311, 2314, 2319, 2321, 2401, 2402, 2403, 2404, 2415, 2420, 2495, 2501, 2520P, 2520R, 2540P, 2540R, 2671, 2701, 2702, 2703, 2955, 3066, 3158, 3210, 3211, 3215, 3277, 3284, 3286, 3330, 3333, 3340, 3410, 3411, 3420, 3505, 3525, 3704, or 3705.

Coding Considerations
For 2741 or 3767 devices (comparably equipped for 2741 operation), specify 2702 or 2703 as the device type.

For the following devices, which have no device type, specify:

- ICA for Integrated Communications Adapter
- CTCA for Channel-to-Channel Adapter

The system console must be specified in both the RDEVICE and RCTLUNIT macros. Specify the system console in both macros as follows:

System/370 Model	System Console
135, 145, 155 II	3210 or 3215
158	3158 (if it is in display mode) or 3215 (if it is in printer-keyboard mode and has the 3213 Printer Model 1)
165 II, 168	3066

The device types, 2540R and 2540P, refer to the same IBM 2540 Card Read Punch (as do 1442P and 1442R, and 2520P and 2520R). Each logical device must be specified in a separate RDEVICE macro.

In addition, any other device that can be attached to a real System/370 can be specified in the RDEVICE macro by its device type. Any device type code that does not appear in the list of supported devices in the "Configurations" section of Part 1 is an unsupported device. For unsupported devices that do not have a device type listed under the DEVTYPE operand, you should code the subclass on the CLASS operand. Then unsupported devices can be dedicated to a virtual machine, and CP can log the error recordings, if there are any. CP does not use unsupported devices for its own operations.

If a device specified in the RDEVICE macro is not supported by VM/370, the following MNOTE message (warning level) is generated:

```
UNSUPPORTED DEVICE TYPE
```

The device is generated as a unsupported device. An unsupported device can only be used if it is dedicated to a virtual machine. It is dedicated to a virtual machine if a DEDICATE control statement is coded in the VM/370 directory for the virtual machine, or if it is attached to it by the CP ATTACH command.

MODEL=model
is the model number, if any, of a 2305, 3330, 3333, 3704, or 3705 device, or a tape device.

model is a value that can be:

1 or 2 for a 2305 device.
1, 2, or 11 for a 3330 device.
1 or 11 for a 3333 device.
1, 2, 3, 4, 5, 6, 7, or 8 for a 3704 or 3705 device.
1, 2, 3, 4, 5, 6, 7, or 8 for a tape device.

FEATURE=(feature[,feature]...)
are the device's optional features. These features can be written in any order. These features are:

Feature	Explanation
7-TRACK	7-track head on a tape drive
CONV	Conversion feature on a 7-track tape drive
DUALDENS	Dual density on a tape drive
OPRDR	Operator identification card reader on a 3277 Model 2
TRANS	Translation feature on a 7-track tape drive
UNVCHSET	Universal character set printer
2CHANSW	Two-channel switch feature for tape or DASD drive
4CHANSW	Four-channel switch feature for tape or DASD drive

Coding Considerations
To allow CMS to correctly verify tape mode set operations, the correct feature code for a tape device must be specified.

If the 3277 display device is equipped with the optional operator identification card reader, then the virtual machine operator can gain access to the system (log on) only if he inserts a magnetically encoded card. Use the FEATURE=OPRDR operand of the RDEVICE macro to specify a 3277 device with a card reader. If you do not want access authorization by a card reader, do not code FEATURE=OPRDR in the RDEVICE macro. FEATURE=OPRDR is invalid if DEVTYPE=3158.

If the hardware configuration includes a Two- or Four-Channel Switch feature, each device on the switchable control unit must be generated with FEATURE=2CHANSW or FEATURE=4CHANSW coded on the RDEVICE macro.

CLASS={(cl[,cl]...) | DASD | TAPE | TERM | GRAF | URI | URO}
is the device class; either the output spooling class or a special subclass for unsupported devices.

Output Spooling Classes
The spooling classes (cl, cl ...) list up to four output spooling classes separated by commas. This form of the CLASS operand can only be specified for a 1403, 1443, or 3211 printer, or 2520P, 2540P or 3525 card punch. The spooling class, cl, is one alphameric character. If you specify more than one class, you must separate them by commas. If no class is specified, class A is assumed for printers and punches.

Coding Considerations: The following information should be helpful when you code this operand:

- For more information about spooling classes, see the VM/370: Operator's Guide.
- The class is used by the CP START command and may be changed by this command. For a complete description of the START command, see the VM/370: Command Language Guide for General Users.
- A class C punch should be included if accounting cards are desired.

Subclass for Unsupported Devices
Specify a device subclass only for unsupported device types. CP uses the subclass when it translates virtual CCW strings directed to unsupported devices. This form of the CLASS operand is valid only if the device type specified on the DEVTYPE operand does not appear in the list of valid device types.

The subclasses are:

DASD Direct Access Storage Devices
TAPE Tape devices
TERM Terminals
GRAF Display mode terminals
URI Unit record input devices
URO Unit record output devices

You must determine the correct subclass to specify for any device type that does not appear in the list of valid device types under the DEVTYPE operand. Do not code a subclass for any device type that appears in that list. For example, a 1287 Optical Reader is an unsupported device for VM/370. It does not appear in the list of supported devices in Part 1 and is not listed as a device type for the DEVTYPE operand of the RDEVICE macro. However, you can define a 1287 and use it if you dedicate it to a virtual machine. You must decide the correct subclass. For example,

```
RDEVICE ADDRESS=010,DEVTYPE=1287,CLASS=URI
```

defines a 1287 Optical Reader at address 010. The 1287 belongs to the unit record input (URI) subclass.

Note: If you use this form of the CLASS operand and the unsupported device does not function properly, try dedicating the device to a virtual=real machine and inhibiting CCW translation (by issuing SET NOTRANS ON). Note that a maximum of twenty-four sense bytes can be contained in the RDEVBLOK created for an unsupported device.

ADAPTER={BSCA | IBM1 | TELE2 | TYPE1 | TYPE2}

is the terminal control or transmission adapter used to connect a telecommunications I/O device to its control unit. This operand is required if a DEVTYPE of 2701, 2702, 2703, 3704, 3705, or ICA is specified, and is ignored if specified for any other device type.

BSCA specifies an IBM Binary Synchronous Terminal Adapter Type II for a 2701, or an IBM Binary Synchronous Terminal Control Type II for a 2703, 3704, or 3705.

IBM1 specifies that an IBM Terminal Adapter Type I attaches a 1050 or 2741 to a 2701, or that an IBM Terminal Control Type I attaches a 1050 or 2741 to a 2702 or 2703, or that a Line Interface Base Type I attaches a 1050 or 2741 to a 3704 or 3705.

TELE2 specifies that a CPT-TWX (Models 33/35) Terminal attaches to a Telegraph Terminal Adapter Type II in a 2701, or to a Telegraph Terminal Control Type II in a 2702 or 2703, or to a Line Interface Base Type I in a 3704 or 3705.

TYPE1 or TYPE2 specifies the channel adapter accessed by a 3704 or 3705. If TYPE2 is specified, CPTYPE should also be specified.

SETADDR=sadnum

is the set address (SAD) command to be issued for a telecommunications line attached to a 2702, 3704, or 3705 control unit. This operand is required if the device is a 2702.

Sadnum

Value	Command
0	SADZERO
1	SADONE
2	SADTWO
3	SADTHREE
4	(no SAD command is issued)

CPTYPE={EP | NCP | PEP}

is the 3704/3705 control program to be run in a 3704 or 3705 Communications Controller.

EP specifies the 2701, 2702, or 2703 Emulation Program.

NCP specifies the Network Control Program.

PEP specifies the Partitioned Emulation Program.

Notes:

1. If you want any of these three control programs to be able to execute, either omit the CPTYPE= operand or specify EP. If CPTYPE=NCP or PEP is specified, an Emulation Program cannot be loaded successfully.
2. If ADAPTER=TYPE2 is specified, you must code the CPTYPE operand.

CPNAME=cpname
is the one- to eight-character name of a 3704/3705 control program that is to be automatically loaded in the 3704 or 3705 at IPL time. If an automatic load is not desired, omit this operand.

MAXDIAL=nn
is a one- or two-digit decimal number indicating the maximum number of 2701, 2702, or 2703 emulation lines that may be active at any one time. The RDEVICE macro generates this number of additional RDEVBLOKs to accommodate these emulation lines.

BASEADD=cuu
is the native address (load address) of the 3704/3705 that controls the physical line(s). This operand is required for correct operation of VM/370 recovery management for emulation lines based on a 3704/3705. This operand is valid only if ADAPTER=IBM1 (or =TELE2 or =BSCA).

CLUSTER=label
is the label of the CLUSTER macro that defines the clustered or standalone station attached to this line.

Examples:

The following examples illustrate the use of the RDEVICE macro instructions to describe a 1403 printer with the Universal Character Set (UCS) feature, four 9-track, 800 bpi tape drives, and eight CPT-TWX lines on a 2702.

```
RDEVICE ADDRESS=00E,DEVTYPE=1403,FEATURE=UNVCHSET,
        CLASS=(A,C)
RDEVICE ADDRESS=(0C0,4),DEVTYPE=2401
RDEVICE ADDRESS=(030,8),DEVTYPE=2702,ADAPTER=TELE2,
        SETADDR=2
```

MNOTE Messages:

The following MNOTE messages may be generated by the RDEVICE macro instruction.

INVALID DEVICE ADDRESS
indicates that one or more of the device addresses coded was specified incorrectly. If DEVTYPE=CTCA, the device address must be in the form cu0, for example, 270.

UNSUPPORTED DEVICE TYPE
indicates that the device specified is not supported by VM/370. It is generated as a non-supported device, and can be used only if it is dedicated to a virtual machine.

MORE THAN 256 DEVICES
indicates that a value greater than 256 was specified in the number parameter of the ADDRESS operand; or that the base address plus the number of devices specified generate an address greater than X'FF'.

INVALID MODEL NUMBER
indicates that the model number specified was not a permitted specification.

INVALID ADAPTER TYPE
indicates that the adapter specified was not a permitted specification, or that the ADAPTER operand was not coded.

INVALID SETADDR VALUE
indicates that the SETADDR value specified is not a permitted specification, or that the SETADDR operand was not coded for a 2702 device.

INVALID FEATURE SPECIFIED
indicates that the feature specified is not a permitted specification.

INVALID CLASS PARAMETER
indicates that an invalid form of the CLASS operand was specified, or that the CLASS operand was specified for a device to which it is not applicable.

RDEVICE MACRO OUT OF SEQUENCE
indicates that the RDEVICE macros are not in their proper position in the real I/O configuration file.

ONLY ONE 3066 OR 3158 CONSOLE ALLOWED
indicates that the ADDRESS=(cuu,nn) operand is invalid. The value nn cannot be greater than one because only one 3066 can be coded on a RDEVICE macro and only one 3158 can be coded on a RDEVICE macro.

| MORE THAN 16 TP CONCENTRATORS AND LINES
| indicates that more than the maximum of 16 binary synchronous lines
| are defined to support remote 3270 configurations.

| INVALID 'CLUSTER' OPERANDS
| indicates that the CLUSTER operand is specified incorrectly; a
| valid assembler language label must be specified.

If any of the above errors (other than UNSUPPORTED DEVICE TYPE) are detected, the real device block is not generated.

Unit Record Error Messages

The RDEVICE macro instruction generates an entry in a table of printers or a table of punches or a table of readers for spooling when DEVTYPE=1403, 3211, 1443, 2540P, 3525, 2540R, 3505, or 2501 is specified. Each table has a maximum of 32 entries; one of the following messages results if more than 32 readers, printers, or punches are specified.

MORE THAN 32 READERS
MORE THAN 32 PRINTERS
MORE THAN 32 PUNCHES

If any of these messages prints, it indicates that the RDEVBLOK is generated, but no entry is made in the printer or punch table; the device cannot be used for CP spooling.

3704/3705 Error Messages

The RDEVICE macro instruction generates an entry in a table of programmable communications controllers when DEVTYPE=3704 or 3705 is specified. This table can have a maximum of 10 entries; the following message results if more than ten 3704 or 3705 devices are specified:

MORE THAN 10 TP CONCENTRATORS

If this message prints, it indicates that the RDEVBLOK is generated, but no entry is made in the Programmable Communications Controller table.

RCTLUNIT MACRO

Use the RCTLUNIT macro to generate a real control unit block (RCUBLOK). One RCTLUNIT macro must be specified for each real control unit. The maximum number of real control units is 511, providing you have enough real storage to hold the real control unit blocks (RCUBLOKs). Control units generally fall into two categories: those supporting eight or fewer devices, and those supporting more than eight devices.

A control unit that supports eight or fewer devices must be assigned an address that is divisible by eight. All devices with an address equal to the control unit's address (the base address) or any of the next seven sequential addresses are mapped to this control unit. For example, devices with addresses of 018 through 01F are mapped to a control unit with address 018.

On a multiplexer channel, several device addresses may fall within the address range of one RCTLUNIT macro. When this occurs, only one RCTLUNIT macro may be coded, even though more than one real control unit is present. For example, a system console at address 009, a 2540 reader at address 00C and a 2540 punch at address 00D would be defined in a single RCTLUNIT macro with a control unit address of 008, even though the system console and the 2540 card reader punch have different real control units. In this case, any valid control unit type can be coded. The only exception to this is that control units that operate on a shared subchannel must be specified by separate RCTLUNIT macros.

For control units supporting a range of more than eight device addresses, use the FEATURE= operand. The base address must be divisible by sixteen. All devices from the base address up to the number of devices specified by the FEATURE= operand are mapped to the specified control unit. When a control unit has the hardware feature that allows it to support more than eight devices, the RCTLUNIT macro must specify FEATURE=xxx-DEVICE, where xxx is the number of addressable devices that can be attached to this control unit. The number of devices specified must be divisible by sixteen and rounded to the next higher increment of sixteen if not divisible. The maximum number of devices that can be attached to a control unit is 256.

For example, if you have a 3830 control unit with the 32-device feature installed, you must specify FEATURE=32-DEVICE for it, even if fewer than thirty-two 3330s are installed.

A device that attaches directly to the channel without a separate control unit must still have a RCTLUNIT macro coded for it. For example, if a 3210 is defined with an RDEVICE macro, it must have a corresponding RCTLUNIT macro.

The RCTLUNIT macro instructions describing the control units for your installation's computing system may be in any order, but they must be contiguous and follow all of the RDEVICE macro instructions in the module DMKRIO. The first RCTLUNIT macro instruction also generates the label DMKRIOCU, which indicates the start of the real control unit blocks to CP.

The name field may not be specified for the RCTLUNIT macro instruction; if a name is specified it is ignored. The macro generates a name by appending the control unit address to the characters RCU. For example, if the control unit address is 230, the name RCU230 is generated.

The format of the RCTLUNIT macro is:

Name	Operation	Operands
	RCTLUNIT	ADDRESS=address
		,CUTYPE=type
		[,FEATURE=xxx-DEVICE]

where:

ADDRESS=address
is the real address of the control unit. The address consists of three hexadecimal digits. The high order digit is the channel address of this control unit. The low order two digits must be the lowest address of the control unit. The first digit may be any hexadecimal number from 0 to F. If the xxx-DEVICE feature is supported, the low order digit must be 0. Otherwise, it may be either 0 or 8.

Note: If your installation has a 2701, 2702, or 2703, and a 3704 or 3705 executing in emulation mode, you must be sure that their addresses are not the same.

CUTYPE=type
is the device type of the control unit. One of the following device type numbers can be specified: 1052, 1442, 2150, 2250, 2314, 2319, 2403, 2404, 2415, 2495, 2501, 2520, 2701, 2702, 2703, 2803, 2804, 2820, 2821, 2822, 2826, 2835, 2840, 2841, 2844, 2845, 2848, 2955, 3066, 3158, 3210, 3215, 3272, 3277, 3345, 3411, 3505, 3525, 3704, 3705, 3803, 3811, 3830, ICA, IFA, ISC, CTCA.

In addition, any other control unit that can be attached to a real System/370 may be specified in an RCTLUNIT macro instruction by its device type.

Even though some devices attach directly to the channel without a separate control unit, an RCTLUNIT macro instruction must be included for them. For example, if you want to define a 3215, you must code an RDEVICE and RCTLUNIT macro for the 3215; even though the 3215 does not require a control unit, it requires a RCTLUNIT macro.

FEATURE=xxx-DEVICE
is the optional control unit feature. The feature, xxx-DEVICE, indicates that the control unit is controlling more than eight devices. The prefix, xxx, can be 16, 32, 48, 64, 80, 96, 112, 128, 144, 160, 176, 192, 208, 224, 240, or 256. "Appendix B: Configuration Aid" lists the maximum number of devices that may be specified for each control unit. This feature may be specified for a 2403, 2702, 2703, 2803, 2835, 3272, 3345, 3704, 3705, 3803, 3830, ICA, or ISC.

For any unsupported control unit, FEATURE=16-DEVICE is valid and is the maximum you can specify. Unsupported control units are any that do not appear in the "Configurations" section of Part 1.

Examples:

The following examples illustrate the use of the RCTLUNIT macro instruction to describe the control units for: a 3215 console printer-keyboard with address 009, a 2314, and a 3705 with lines 040 through 04B.

```
RCTLUNIT ADDRESS=008,CUTYPE=3215
RCTLUNIT ADDRESS=230,CUTYPE=2314
RCTLUNIT ADDRESS=040,CUTYPE=3705,FEATURE=16-DEVICE
```

MNOTE Messages:

The following MNOTE messages may be generated by the RCTLUNIT macro instruction.

INVALID DEVICE CONFIGURATION
: indicates that CUTYPE=CTCA was specified, but there was more than one RDEVICE macro defining devices on the control unit.

INVALID CONTROL UNIT ADDRESS
: indicates that either the channel or control unit portion of the address is not in the range 0 to F, or that the last digit of the address is neither 0 nor 8, or that the control unit portion plus the number of devices specified is greater than 256.

INVALID CONTROL UNIT ADDRESS FOR SPECIFIED FEATURE
: indicates that FEATURE=xxx-DEVICE is specified but that the last digit of the control unit address is not 0.

INVALID CONTROL UNIT TYPE FOR SPECIFIED FEATURE
: indicates that FEATURE=xxx-DEVICE is specified but the type number is not valid for the specified feature.

INVALID FEATURE SPECIFIED
: indicates that the argument of the FEATURE= keyword is not 16-DEVICE to 256-DEVICE in increments of 16.

RCTLUNIT MACRO OUT OF SEQUENCE
: indicates that the RCTLUNIT macros are not in their prcper position in the real I/O configuration file.

If any of the above errors are detected, the real control unit block is not generated. This results in undefined symbols in the real device blocks for this control unit.

RCHANNEL MACRO

Use the RCHANNEL macro to generate a real channel block (RCHBLOK). An RCHANNEL macro instruction must be coded to define each real channel in the I/O configuration.

The RCHANNEL macro instructions describing the channels for your installation may be in any order, but they must be contiguous and follow all of the RCTLUNIT macro instructions in the module DMKRIO. The first RCHANNEL macro instruction also generates the label DMKRIOCH, which indicates the start of the real channel blocks to CP.

No name is specified for the RCHANNEL macro instruction; if a name is specified, it is ignored. The RCHANNEL macro generates a name by appending the channel address to the characters RCHAN. For example, if the channel address is 2, the name RCHAN2 is generated.

The format of the RCHANNEL macro is:

```
┌──────────────────────────────────────────────────────────────────────┐
| Name  |  Operation  |  Operands                                      |
|-------|-------------|------------------------------------------------|
|       |  RCHANNEL   | ADDRESS=address                                |
|       |             | ,CHTYPE=(SELECTOR    )                         |
|       |             |         (MULTIPLEXOR )                         |
|       |             |         (BLKMPXR     )                         |
└──────────────────────────────────────────────────────────────────────┘
```

where:

ADDRESS=address
is the real address of the channel. It is a hexadecimal number from 0 to F.

CHTYPE=(SELECTOR)
(MULTIPLEXOR)
(BLKMPXR)
is the type of channel.

SELECTOR indicates a selector channel.

MULTIPLEXOR indicates a byte-multiplexer channel.

BLKMPXR indicates a block multiplexer channel.

Examples:

The following examples illustrate the use of the RCHANNEL macro instruction to describe a multiplexer channel whose address is 0, a selector channel whose address is 1, and a block multiplexer channel whose address is 2.

```
RCHANNEL  ADDRESS=0,CHTYPE=MULTIPLEXOR
RCHANNEL  ADDRESS=1,CHTYPE=SELECTOR
RCHANNEL  ADDRESS=2,CHTYPE=BLKMPXR
```

MNOTE Messages:

The following MNOTE messages may be generated by the RCHANNEL macro instruction:

INVALID CHANNEL ADDRESS
: indicates that the channel address specified is not in the range 0-F.

INVALID CHANNEL TYPE
: indicates that the channel type specified is neither MULTIPLEXOR, BLKMPXR, nor SELECTOR.

RCHANNEL MACRO OUT OF SEQUENCE
: indicates that the RCHANNEL macros are not in their proper position in the real I/O Configuration file.

If any of the above errors is detected, the real channel block is not generated. This results in undefined symbols in the real control unit blocks for this channel.

RIOGEN MACRO

Use the RIOGEN macro instruction to generate the channel index table and unit record and console tables. It must appear as the last macro instruction before the END statement in the DMKRIO file.

The name field must not be specified for the RIOGEN macro. The format of the RIOGEN macro is:

Name	Operation	Operands
	RIOGEN	CONS=xxx [,ALTCONS=xxx]

where:

CONS=address
is the address of the VM/370 primary system console. The address is a hexadecimal device address that was previously specified in an RDEVICE macro entry. This device must be either a 3066, 3210, 3215, or 1052 (via a 2150 freestanding console), a System Console for the 3158 (in printer-keyboard mode with the 3213 Printer Model 1 required, or in display mode), 7412, or a 3277 (local attachment).

ALTCONS=address
is the address of the alternate console. The address is a hexadecimal device address that was previously specified in an RDEVICE macro instruction. This device, which should be located as close as possible to the primary system console, may be any device supported as a VM/370 logon device (except for those remote terminals connected via 3704/3705 Communication Controllers). This console is used if the primary system console is unavailable at initial program load time. If ALTCONS is not coded or the alternate console is not available, CP enters a disabled wait state with a wait state code of X'005' in the instruction address register (IAR).

Coding Considerations: The alternate console must not be a telecommunications line on a real IBM 3704/3705 Communications Controller.

If the alternate console is an IBM 2741 Communication Terminal, or 3767 Communication Terminal (operating as a 2741), it must use the EBCDIC transmission code.

Examples:

The following examples define a primary system console (01F) with an alternate console (050), and a system console (009) with no alternate console.

```
RIOGEN CCNS=01F,ALTCONS=050
RIOGEN CONS=009
```

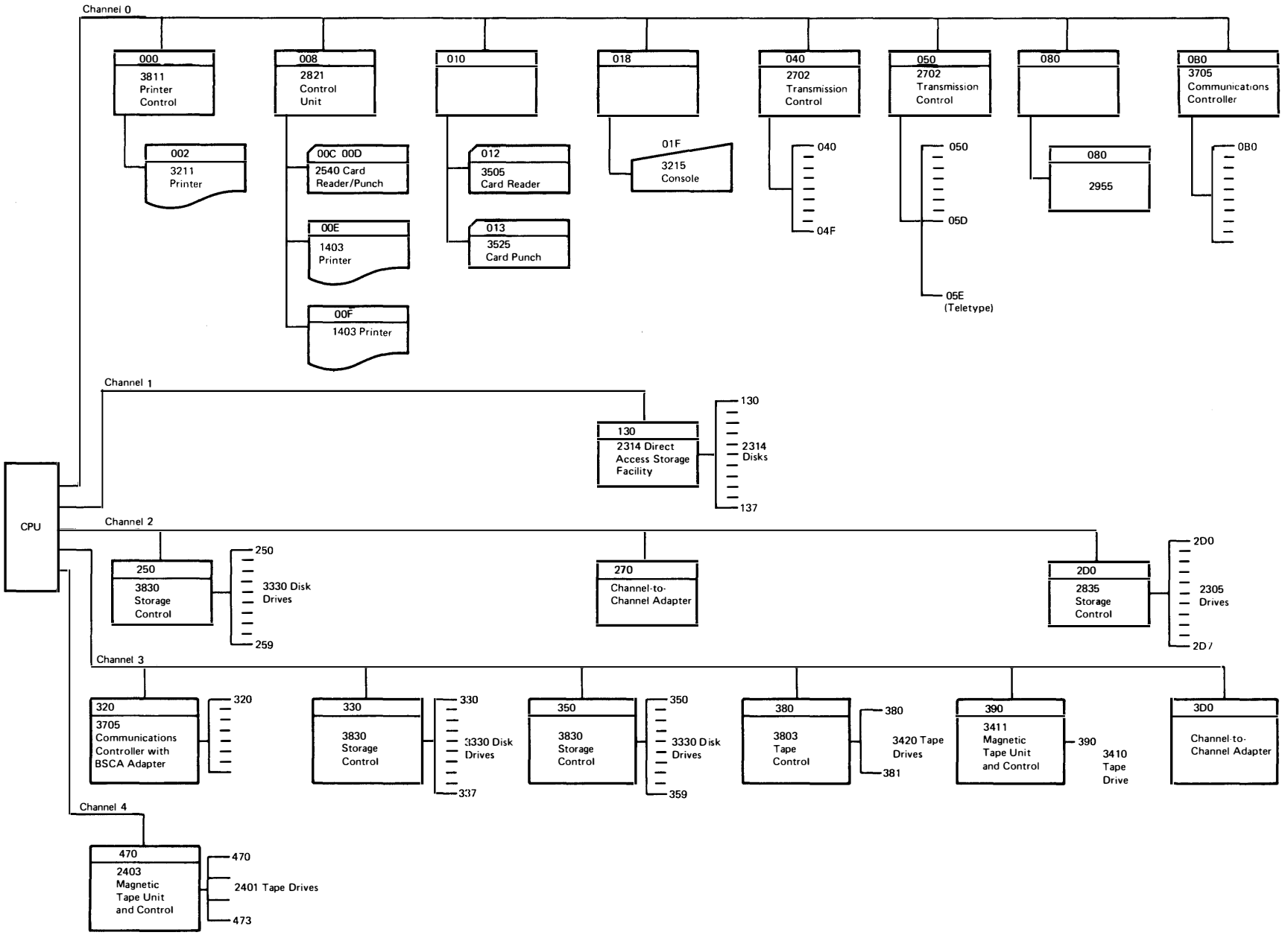
EXAMPLE OF CODING REAL I/O CONFIGURATION FILE (DMKRIO)

In this example, macros are coded to support the following real devices:

- 1 2540 Card Reader/Punch
- 1 3505 Card Reader
- 1 3525 Card Punch
- 2 1403 Printers with the Universal Character Set feature
- 1 3211 Printer with the Universal Character Set feature
- 1 3215 Console Printer-Keyboard
- 1 2955
- 2 3705 Communications Controllers (one with an IBM1 adapter and the other with a BSCA adapter)
- 2 2702 Transmission Control Units (with one that supports Teletype terminals)
- 1 2314 Direct Access Storage Facility with 8 modules
- 1 2305 Fixed Head Storage with 8 drives
- 2 3330 Disk Storage (One unit has eight modules and the other has ten. The unit with ten modules has eight of them switchable between two channels.)
- 1 2401 Magnetic Tape Unit with 4 tape drives
- 1 3410 Magnetic Tape Unit, Model 3, with 1 tape drive
- 1 3420 Magnetic Tape Unit, Model 7, with 2 tape drives
- 1 Multiplexer channel
- 4 Block multiplexer channels
- 1 Channel-to-Channel Adapter

Figure 16 shows the real configuration.

Figure 16. Example of a Real Configuration



The real I/O configuration file that supports this example is:

```
DMKRIO CSECT
       RDEVICE ADDRESS=00C,DEVTYPE=2540R
       RDEVICE ADDRESS=00D,DEVTYPE=2540P,CLASS=(X,A)
       RDEVICE ADDRESS=00E,DEVTYPE=1403, CLASS=(X,A),FEATURE=UNVCHSET
       RDEVICE ADDRESS=00F,DEVTYPE=1403, CLASS=(S),FEATURE=UNVCHSET
       RDEVICE ADDRESS=002,DEVTYPE=3211, CLASS=(X,A),FEATURE=UNVCHSET
       RDEVICE ADDRESS=012,DEVTYPE=3505
       RDEVICE ADDRESS=013,DEVTYPE=3525,CLASS=(X,A)
       RDEVICE ADDRESS=01F,DEVTYPE=3215
       RDEVICE ADDRESS=080,DEVTYPE=2955
       RDEVICE ADDRESS=0B0,DEVTYPE=3705,ADAPTER=IBM1
       RDEVICE ADDRESS=(130,8),DEVTYPE=2314
       RDEVICE ADDRESS=270,DEVTYPE=CTCA
       RDEVICE ADDRESS=320,DEVTYPE=3705,ADAPTER=BSCA
       RDEVICE ADDRESS=3D0,DEVTYPE=CTCA
       RDEVICE ADDRESS=2D0,DEVTYPE=2305,MODEL=2
       RDEVICE ADDRESS=(250,10),DEVTYPE=3330,FEATURE=2CHANSW
       RDEVICE ADDRESS=(350,8),DEVTYPE=3330
       RDEVICE ADDRESS=(358,2),DEVTYPE=3330,MODEL=11
       RDEVICE ADDRESS=(470,4),DEVTYPE=2401,FEATURE=DUALDENS,MODEL=5
       RDEVICE ADDRESS=(040,16),DEVTYPE=2702,ADAPTER=IBM1,SETADDR=2
       RDEVICE ADDRESS=(050,14),DEVTYPE=2702,ADAPTER=IBM1,SETADDR=2
       RDEVICE ADDRESS=05E,DEVTYPE=2702,ADAPTER=TELE2,SETADDR=1
       RDEVICE ADDRESS=(330,8),DEVTYPE=3330,FEATURE=2CHANSW
       RDEVICE ADDRESS=390,DEVTYPE=3410,FEATURE=DUALDENS,MODEL=3
       RDEVICE ADDRESS=(380,2),DEVTYPE=3420,FEATURE=DUALDENS,MODEL=7
       RCTLUNIT ADDRESS=000,CUTYPE=3811
       RCTLUNIT ADDRESS=008,CUTYPE=2821
       RCTLUNIT ADDRESS=0B0,CUTYPE=3705,FEATURE=16-DEVICE
       RCTLUNIT ADDRESS=270,CUTYPE=CTCA
       RCTLUNIT ADDRESS=320,CUTYPE=3705
       RCTLUNIT ADDRESS=3D0,CUTYPE=CTCA
       RCTLUNIT ADDRESS=010,CUTYPE=3505
       RCTLUNIT ADDRESS=018,CUTYPE=3215
       RCTLUNIT ADDRESS=040,CUTYPE=2702,FEATURE=16-DEVICE
       RCTLUNIT ADDRESS=050,CUTYPE=2702,FEATURE=16-DEVICE
       RCTLUNIT ADDRESS=080,CUTYPE=2955
       RCTLUNIT ADDRESS=130,CUTYPE=2314
       RCTLUNIT ADDRESS=470,CUTYPE=2403
       RCTLUNIT ADDRESS=2D0,CUTYPE=2835
       RCTLUNIT ADDRESS=250,CUTYPE=3830,FEATURE=16-DEVICE
       RCTLUNIT ADDRESS=350,CUTYPE=3830,FEATURE=16-DEVICE
       RCTLUNIT ADDRESS=330,CUTYPE=3830
       RCTLUNIT ADDRESS=380,CUTYPE=3803
       RCTLUNIT ADDRESS=390,CUTYPE=3411
       RCHANNEL ADDRESS=0,CHTYPE=MULTIPLEXOR
       RCHANNEL ADDRESS=1,CHTYPE=BLKMPXR
       RCHANNEL ADDRESS=2,CHTYPE=BLKMPXR
       RCHANNEL ADDRESS=3,CHTYPE=BLKMPXR
       RCHANNEL ADDRESS=4,CHTYPE=BLKMPXR
       RIOGEN CONS=01F,ALTCONS=050
       END
```

The CP system control file consists of macros that describe the CP system residence device, the system storage size, the CP-owned direct access devices, the system operator's user identification, and the proper system timer value. The file should be placed in the card reader in the order shown:

```
DMKSYS  CSECT
        SYSOWN  macro
        SYSRES  macro
        SYSOPR  macro
        SYSCOR  macro
        SYSTIME macro
        SYSLOCS macro
        END
```

Note: Samples of the DMKSYS and DMKSNT files are provided with the starter system. If you use these, you can save the CMS system at the end of the system generation procedure. VM/370 prompts you to tell it if you want the sample DMKSYS file punched. You may modify this file if you wish. For example, you could modify the starter system DMKSYS file to add other CP-owned volumes.

The DMKSYS module supplied with the 2314 starter system is:

```
DMKSYS CSECT
       SYSOWN (VMREL2,TEMP)
       SYSRES SYSVOL=VMREL2,SYSRES=131,SYSTYPE=2314,SYSNUC=3,          X
              SYSWRM=6,SYSERR=7
       SYSCOR RMSIZE=512K
       SYSOPR SYSOPER=OPERATOR,SYSDUMP=OPERATOR
       SYSTIME ZONE=4,LOC=WEST,ID=EDT
       SYSLOCS
       END
```

The DMKSYS module supplied with the 3330 starter system is:

```
DMKSYS CSECT
       SYSOWN (VMREL2,TEMP)
       SYSRES SYSVOL=VMREL2,SYSRES=131,SYSTYPE=3330,SYSNUC=2,          X
              SYSWRM=4,SYSERR=5
       SYSCOR RMSIZE=512K
       SYSOPR SYSOPER=OPERATOR, SYSDUMP=OPERATOR
       SYSTIME ZONE=4,LOC=WEST,ID=EDT
       SYSLOCS
       END
```

The DMKSYS module supplied with the 3340 starter system is:

```
DMKSYS CSECT
       SYSOWN (VMREL2,TEMP)
       SYSRES SYSVOL=VMREL2,SYSRES=131,SYSTYPE=3340,SYSNUC=3,          X
              SYSWRM=8,SYSERR=9
       SYSCOR RMSIZE=512K
       SYSOPR SYSOPER=OPERATOR,SYSDUMP=OPERATOR
       SYSTIME ZONE=4,LOC=WEST,ID=EDT
       SYSLOCS
       END
```

PERFORMANCE CONSIDERATIONS FOR CODING THE DMKSYS FILE MACROS

The following recommendations may help reduce arm and channel contention and may improve the performance of a VM/370 system.

- Provide separate CP volumes for paging and spooling and have the volumes mounted on separate channels.
- If you have a heavy I/O production virtual machine (for example, one that is executing OS/VS1 or DOS/VS), try to keep all its major I/O devices on a separate channel from a channel handling the CMS system residence volume or other user's disks.
- Try to keep read-only minidisks (for example, the CMS system residence disk, source disks) that are frequently accessed on separate volumes from users' read/write minidisks. If possible, also keep them on separate channels.
- If your installation is likely to have a large number of CMS users active at one time, you should distribute the CMS activity over two volumes by (1) setting up a second CMS system residence volume and dividing the users between the two CMS system residence volumes or (2) putting your program products on one spindle and the CMS non-resident commands on another spindle.

SYSOWN MACRO

Use the SYSOWN macro to generate the list of CP-owned DASD volumes. A CP-owned volume is either the CP system residence volume, or a volume that contains VM/370 paging, spooling, or temporary disk space. It must contain a CP allocation table at cylinder 0, record 4 allocating these areas. Even if a volume has a VM/370 allocation table at cylinder 0, record 4, allocation data is ignored unless the volume appears as an operand in the SYSOWN macro instruction.

Special Considerations for Allocating Space on CP-Owned Volumes: The following considerations should help you to allocate space efficiently on CP-owned volumes:

- If a volume is specified in a SYSOWN statement but is not mounted when the generated system is loaded (via the IPL command), that volume is considered unavailable to VM/370. Processing continues, if possible. The operator can mount and attach the volume later, if it is needed.

- Only those volumes that contain paging and spooling space or TDSK space need be identified as CP-owned volumes. All other volumes are described either by directory entries or by logically attaching the entire device.

- If you add another volume to the SYSOWN list, you must add it at the end of the list. (Otherwise, if you attempt a warm start after regenerating and loading CP, the relative entry number used to locate system spool buffers is incorrect.) Then reassemble DMKSYS, rebuild the CP nucleus, and reload it on the system residence volume. Use the GENERATE EXEC procedure to reassemble DMKSYS and reload the CP nucleus.

- If your installation has saved systems (systems that can be loaded by name, thus bypassing the initial program load procedure), you must reserve space on a CP-owned volume to hold the named systems you want saved. The DASD space you reserve, for each named system you wish to save, should be enough to contain the number of pages specified in the SYSPGCT operand of the NAMESYS macro, plus one page for system use.

- If your VM/370 system has a 3704 or 3705, you must reserve space on a CP-owned volume to contain the 3704/3705 control program image. See the "Generating a VM/370 System that Supports the 3704/3705" section of Part 1 for information about how much DASD space you should reserve.

The name field must not be specified for the SYSOWN macro.

The format of the SYSOWN macro is:

Name	Operation	Operands
	SYSOWN	(volid [,TEMP] [,PAGE]) [,(volid [,TEMP] [,PAGE])...]

where:

volid
: is the CP-owned volume identification, up to six alphameric characters long.

```
┌     ┐
|,TEMP|
|,PAGE|
└     ┘
```

: indicates to VM/370 how allocatable space on the specified volume should be used.

  TEMP specifies that this volume is to be used for temporary and spool space allocation, and should only be used for paging space if all volumes normally used for paging allocation are full or unavailable. TEMP is the default option. (TEMP space must be formatted by the CP Format/Allocate service program.)

  PAGE specifies that this volume is to be used for paging and temporary disk allocation only.

Example:

The following SYSOWN macro designates the CPDRM1 volume as paging space and the CPDSK1 and CPDSK2 volumes as temporary space:

```
SYSOWN (CPDRM1,PAGE),(CPDSK1,TEMP),CPDSK2
```

MNOTE Messages:

The following MNOTE messages may be generated by the SYSOWN macro instruction:

INVALID PREFERENCE OPTION
: indicates that the second positional sublist parameter following volid is not TEMP or PAGE.

MORE THAN 255 VOLUMES SPECIFIED
: indicates that more than 255 CP-owned volumes are specified in the SYSOWN macro instruction.

DUPLICATE VOLUME SERIAL SPECIFIED
: indicates that the same volume serial number is specified more than once.

NO VOLUMES SPECIFIED
: indicates that the SYSOWN macro instruction is specified without any positional parameters.

WARNING - NO TEMP SPACE SPECIFIED
: indicates that no volume is specified as being preferred for TEMP space allocation. This means that no spooling operations may be performed. Thus any data transfer channel program started to a virtual unit record output device ends with UC status in the virtual CSW and the intervention required bit set in the virtual sense byte.

SYSRES MACRO

Use the SYSRES macro instruction to describe the characteristics of the CP system residence volume.

The name field must not be specified for the SYSRES macro instruction.

Special Considerations for Coding the SYSRES Macro: The following information should help you when you code the SYSRES macro:

- The cylinders required for SYSNUC, SYSERR, and SYSWRM must be formatted using the CP FORMAT service program, and must be allocated as permanent space on the SYSRES volume, but not in cylinder 0.
- VM/370 allows the 2314 or 2319 "alternate tracks" cylinders 200-203, 3340 Model 35 alternate cylinder 348, and 3340 Model 70 alternate cylinders 696-697, to be used for normal data if they are not needed to replace defective tracks.

The format of the SYSRES macro is:

```
| Name | Operation | Operands                                          |
|------|-----------|---------------------------------------------------|
|      | SYSRES    | SYSVOL=serial,                                    |
|      |           | SYSRES=address,                                   |
|      |           | SYSTYPE={2314},                                   |
|      |           |         {2319}                                    |
|      |           |         {3330}                                    |
|      |           |         {3340}                                    |
|      |           |         {2305}                                    |
|      |           | SYSNUC=strtcyl,                                   |
|      |           | SYSERR=cylinder,                                  |
|      |           | SYSWRM=(strtcyl,cylcount)                         |
```

where:

SYSVOL=serial
: is the volume identification of the system residence disk. The volume serial number, serial, is a character string with a maximum length of 6 characters.

SYSRES=address
: designates the device address of the DASD to contain the newly-generated system. This address is used only when the VM/370 nucleus is generated and written on the disk. Thereafter, you can IPL the system from any addressable disk device.

  The address is a three-digit hexadecimal device address.

SYSTYPE={2314 | 2319 | 3330 | 3340 | 2305}
: is the device type of the system residence device.

SYSNUC=strtcyl
: is the number of the real starting cylinder where the CP nucleus resides.

The cylinder, strtcyl, is a one- to three-digit decimal number.

Normally, a 2314 or 2319 device requires three contiguous cylinders, a 2305 or 3340 device requires four contiguous cylinders, and a 3330/3333 device requires two contiguous cylinders to contain the CP nucleus.

Systems being generated with the virtual=real option require additional space. For information about how much space to allocate for virtual=real configurations, see the "Specifying a Virtual=Real Machine" section.

SYSERR=cylinder
is the number of the real starting cylinder where error recording records are written. Two contiguous cylinders are required for error recording.

The cylinder is a one- to three-digit decimal number.

SYSWRM=[(]strtcyl[,cylcount[)]]
[,1]

is the number of the real starting cylinder, and optionally the maximum number of cylinders, to contain the warm start data.

The strtcyl is a one- to three-digit decimal number designating the first real cylinder where CP warm start information is to be saved.

The cylcount value is a one-digit decimal number (1 through 9) that defines the maximum number of cylinders to contain warm start data. If cylcount is not specified, 1 is the default value.

The cylcount operand is optional; if included, the strtcyl and cylcount operands must be separated by a comma and enclosed in parentheses. Parentheses are optional when only the strtcyl operand is specified. The following are valid entries for one cylinder warm start areas:

```
SYSWRM=(202,1)
SYSWRM=(202)
SYSWRM=202
```

Use the following formulas to calculate the number of warm start cylinders required.

Device Type	Formula
2314/2319	$N=\left[\frac{34 + (NSF/40) + (NCS/510) + ((NAU \times 4)/50)}{32}\right]$
3340/2305	$N=\left[\frac{26 + (NSF/40) + (NCS/510) + ((NAU \times 4)/50)}{24}\right]$
3330	$N=\left[\frac{59 + (NSF/40) + (NCS/510) + ((NAU \times 4)/50)}{57}\right]$

where:

N is the number of cylinders required for warm start data.

NSF is the maximum number of spool files in the system at any one time. There are 40 spool file blocks per 4096-byte record.

NSC is the number of cylinders available for spool files. There are 510 allocation blocks per 4096-byte record.

NAU is the maximum number of active users in the system at any one time. There are 50 accounting records per 4096-byte record.

Example:

The following SYSRES macro defines the system residence volume as the 2314 volume with a serial number of CPDSK1. During the system generation procedure this volume is found at address 230. The VM/370 system starts at cylinder 198, the error recording area starts at cylinder 4, and the warm start storage area is cylinder 202. The format of the SYSRES macro is:

```
SYSRES SYSVOL=CPDSK1,SYSRES=230,SYSTYPE=2314,SYSNUC=198,          X
       SYSERR=4,SYSWRM=(202,1)
```

MNOTE Messages:

The following MNOTE messages may be generated by the SYSRES macro instruction:

INVALID DEVICE TYPE FOR SYSRES
: indicates that the argument of the SYSTYPE operand is not 2305, 2314, 2319, 3330, or 3340.

SYSVOL NOT IN OWNED LIST
: indicates that the argument of the SYSVOL operand is a volume serial number that was not previously specified in the SYSOWN macro instruction. This error occurs if the SYSOWN macro instruction does not appear in an assembly before the SYSRES macro instruction.

SYSOPR MACRO

Use the SYSOPR macro instruction to specify the system operator's userid, and the userid of the operator who is to receive VM/370 system dumps. The same userid may be specified in both operands.

The name field must not be specified for the SYSOPR macro instruction.

The format of the SYSOPR macro is:

Name	Operation	Operands
	SYSOPR	[SYSOPER=OPERATOR] SYSOPER=userid
		[,SYSDUMP=OPERATNS] ,SYSDUMP=userid

where:

[SYSOPER=OPERATOR]
SYSOPER=userid

is the userid of the virtual machine to be assigned to the system operator. If SYSOPER is not specified, the userid "OPERATOR" is used.

The userid is a character string up to eight characters long.

[SYSDUMP=OPERATNS]
SYSDUMP=userid

is the userid of the virtual machine whose spool input receives the system dump file after a system restart. If SYSDUMP is not specified, the userid "OPERATNS" is used.

The userid is a character string up to eight characters long.

Example:

The following SYSOPR macro designates the OP virtual machine as the system operator and directs the system dumps to the CPSYS virtual machine.

```
SYSOPR SYSOPER=OP,SYSDUMP=CPSYS
```

SYSCOR MACRO

Use the SYSCOR macro instruction to generate the internal control block called the CORTABLE.

The name field must not be specified for the SYSCOR macro instruction.

The format of the SYSCOR macro is:

Name	Operation	Operands
	SYSCOR	RMSIZE={xxxxxK / yyM}

where:

RMSIZE={xxxxxK / yyM}

is the amount of real storage available for VM/370. This value limits the amount of real storage used by VM/370 if it is less than the total amount of real storage available in the real machine. If the available real storage is less than this value when VM/370 is initialized, a message indicating the amount of storage available is displayed at the operator's console.

The value, xxxxx, is a three- to five-digit number that denotes the amount of real storage in terms of K bytes, where 1K=1024 bytes. This value may range from 240K to 16384K. It must always be a multiple of 2.

The value, yy, is a one- or two-digit number that denotes the amount of storage in terms of M bytes, where 1M=1024K bytes. This value may range from 1M to 16M.

Note: Do not specify a value substantially larger than the size of real storage, because the core table generated uses a large amount of real storage.

Examples:

The first example defines real storage as 256K (272,144) bytes and the second example defines real storage as 1M (1,048,576) bytes.

```
SYSCOR  RMSIZE=256K
SYSCOR  RMSIZE=1M
```

MNOTE Messages:

The following MNOTE messages may be generated by the SYSCOR macro instruction:

INVALID RMSIZE SPECIFIED
: indicates that the real machine size specified is less than 240K, greater than 16M, or not a multiple of 2K.

SYSTIME MACRO

Use the SYSTIME macro instruction to generate information needed to set the hardware time of day (TOD) clock. The value stored in the TOD clock represents time taken at Greenwich Mean Time, and must be corrected to local time whenever it is examined. The system operator can alter the defined time value by using the store clock function.

The name field must not be specified for the SYSTIME macro instruction.

The format of the SYSTIME macro is:

Name	Operation	Operands
	SYSTIME	[ZONE=0]
		ZONE=h
		ZONE=(h,m)
		ZONE=(h,m,s)
		ZONE=(h,,s)
		[,LOC=EAST]
		,LOC=WEST
		[,ID=GMT]
		,ID=xxx

where:

[ZONE=0]
ZONE=h
ZONE=(h,m)
ZONE=(h,m,s)
ZONE=(h,,s)

is the time zone differential from Greenwich Mean Time. If ZONE is not specified, a value of 0 hours (Greenwich Mean Time) is used.

The variable, h, is a number that represents hours. It can have a value from 0 to 13, but when coupled with the m and s fields, the total effective zone differential must not exceed 13 hours.

The variable, m, is a number that represents minutes.

The variable, s, is a number that represents seconds.

[LOC=EAST]
LOC=WEST

specifies whether the time zone differential is to be taken EAST or WEST of Greenwich Mean Time. The default value for LOC is EAST. When the effective value of ZONE is 0, the setting of LOC is meaningless.

```
┌      ┐
|ID=GMT|
|ID=xxx|
└      ┘
```

is the name of the time zone. The default for ID is GMT. The variable, xxx, is a three-character string.

Examples:

The following examples show how to code the SYSTIME macro for several different time zones.

```
SYSTIME ZONE=5,LOC=WEST,ID=EST (Eastern Standard Time)
SYSTIME ZONE=4,LOC=WEST,ID=EDT (Eastern Daylight Time)
SYSTIME ZONE=6,LOC=WEST,ID=CST (Central Standard Time)
SYSTIME ZONE=7,LOC=WEST,ID=MST (Mountain Standard Time)
SYSTIME ZONE=1,LOC=EAST,ID=SET (Standard European Time)
SYSTIME ZONE=1,LOC=EAST,ID=BST (British Summer Time)
```

MNOTE Messages:

The following MNOTE messages may be generated by the SYSTIME macro instruction:

INVALID LOC SPECIFIED
: indicates that the argument of the LOC operand is not EAST or WEST.

ZONE GREATER THAN 13 HOURS
: indicates that the total value of the hours, minutes, and seconds subfields of ZONE is greater than 13 hours.

ZERO LENGTH ID SPECIFIED
: indicates that a zero length ID is specified.

INVALID ZONE
: indicates that either the hours subfield of ZONE is missing, or more than three subfields are specified.

SYSLOCS MACRO

The SYSLOCS macro instruction is a required macro used to generate internal pointer variables. This must be the last macro in the DMKSYS deck.

This macro is required and must be the last macro in the DMKSYS file.

The name field must not be specified for the SYSLOCS macro instruction. No operands are required for the SYSLOCS macro; if one is specified, it is ignored.

The format of the SYSLOCS macro is:

Name	Operation	Operands
	SYSLOCS	

Example:

An example of the SYSLOCS macro is:

```
     SYSLOCS
```

The VM/370 directory contains the entries of all potential virtual machines that are permitted to log on the VM/370 system. Without the proper directory entry, a user cannot log on to VM/370. The entries in the directory contain the user identification and password, the virtual machine I/O configuration, associated virtual and real addresses, disk usage values, virtual CPU storage size, and other options. Each user in the directory, except those whose password is NOLOG, must have at least one device. Any of the various devices described meet this requirement; for example, the device may be a console or a spool device. These options are discussed in the directory program control statement descriptions.

The VM/370 directory usually resides on the VM/370 system residence disk, and is pointed to by the VOL1 label (cylinder 0, track 0, record 3). The VM/370 Directory program (module DMKDIR, invoked by the DIRECT command, or run standalone) processes the control statements you prepare and writes the VM/370 directory on disk. You already described your installation's real configuration when you created the real I/O configuration file. Now, you describe the many virtual configurations for your installation with the Directory program control statements.

To create a VM/370 directory, you must:

- Prepare the Directory program control statements
- Format and allocate the DASD space to contain the VM/370 directory
- Execute the Directory program

At this time, you should prepare the Directory program control statements. Later, during the system generation procedure, you must (1) format and allocate DASD space for the VM/370 directory and (2) generate it. The step-by-step description of the system generation procedure that is in Part 3 of this manual reminds you to create your VM/370 directory.

CONSIDERATIONS FOR PREPARING THE DIRECTORY CONTROL STATEMENTS

First, prepare a directory control statement that defines the device on which the VM/370 directory is to be written. This statement (DIRECTORY) must be the first control statement in the input to the Directory program, and is followed by the sets of statements describing your installation's virtual machines.

Next, prepare Directory program control statements describing each virtual machine in your installation. The descriptions contain accounting data, options, and virtual machine configurations for each virtual machine that appears in the VM/370 directory. Information about coding these control statements is found in the section, "The Directory Program." Appendix C contains an example of a basic set of VM/370 directory entries.

If you intend to define more than 73 virtual devices for a single virtual machine, be aware that any single request for free storage in excess of 512 doublewords (a full page) causes the VM/370 system to abnormally terminate (abend code PTR007) if the extra storage is not

available on a contiguous page. Therefore, two contiguous pages of free storage must be available in order to log on a virtual machine with more than 73 virtual devices (three contiguous pages for a virtual machine with more than 146 virtual devices, and so on). Contiguous pages of free storage are sure to be available only immediately after IPL, before other virtual machines have logged on. Therefore, a virtual machine with more than 73 devices should be the first to log on after IPL.

VM/370 does not check for overlapping extents; therefore, you must ensure that minidisk extents defined in the VM/370 directory do not overlap.

You must define one or more virtual machines for the operator and should define virtual machines for the system analyst or system programmer.

The operator's virtual machines should be able to control:

- The VM/370 sessions
- Allocation of machine resources
- Spooling activity
- Online disk areas

You should also define virtual machines for system analysts that are equipped to:

- Perform system analysis
- Modify certain VM/370 functions

and additional virtual machines to update or operate:

- The CP system
- The CMS system
- The RSCS system, if you generate one
- The hardware
- Other operating systems that run in the virtual machine environment
- The Installation Verification Procedure

SYSTEM SUPPORT VIRTUAL MACHINES

At system generation time, two additional virtual machines should be created beyond those needed by normal users (one each for hardware and software support). The IBM FE Program Systems Representative should be consulted when the configurations for these virtual machines are being determined.

Hardware Support

The hardware support is for:

- The CPU, which must be supported in a dedicated environment because there is no method currently available that allows concurrent support of the CPU, real storage, or channels when executing problem programs.

- The input/output equipment, which can be supported using online test (OLT) under OLTSEP. The OLTSEP program can be executed in its own virtual machine.

Any of the offline testing capabilities of the system devices can be used on inactive units while the system is operating.

To perform online hardware support, a virtual machine must be defined in the VM/370 directory for the IBM service representative. The virtual machine should have enough virtual storage defined to execute OLTSEP. Normally, the service representative requires that the device being tested be dedicated to his virtual machine. (The system operator can dedicate devices to a virtual machine by issuing the ATTACH command.)

Also the virtual machine for hardware support should have the minimum configuration required to run online tests, and provide access to CMS with a read/write minidisk. Privilege class F should be assigned to allow the hardware diagnostics to be run, and error recording and retrieval facilities to be utilized.

The hardware service representative's virtual machine should also have access to CMS and to the error recording area of the system residence volume. An EREP program runs under CMS thus allowing editing and printing of all VM/370-recorded machine check and channel check errors.

An example of a virtual machine for hardware support (for a 2314 system) is:

```
USER CE CE 320K 1M EFG
 ACCOUNT ACT2 CE
 CONSOLE 009 3215
 SPOOL 00C 2540 READER A
 SPOOL 00D 2540 PUNCH A
 SPOOL 00E 1403 A
 MDISK 191 2314 014 005 CPV2L0 WR READ WRITE
 LINK MAINT 194 194 RR
 LINK MAINT 190 190 RR
```

Virtual device 191 is a CMS A-disk that is to contain programs required by the service representative. Input/output units to be tested are normally attached to the virtual machine for the duration of the test. Privilege class F is provided to allow the service representative to specify an intensive error recording mode for a particular device being tested, and to examine or clear the system error recording area.

This directory entry is included in the VM/370 directory provided with the starter system.

Software Support

The virtual machine for software support should have the minimum configuration necessary to create (virtually) problems that occur on the real machine. The ECMODE option must be specified in the directory OPTION control statement for this machine. You should assign privilege class G to the user of this machine. Also, you should assign privilege class E if he is to examine real storage addresses, and privilege class B if he is to allocate devices.

An example of a virtual machine for software support (for a 2314 system) is:

```
USER MAINT CPCMS 512K 1M BCEG
 ACCOUNT ACT3 MAINT
 OPTION ECMODE REALTIMER
 CONSOLE 009 3215
```

```
SPOOL 00C 2540 READER A
SPOOL 00D 2540 PUNCH A
SPOOL 00E 1403 A
MDISK 190 2314 035 110 CPV2L0 MR READ
MDISK 191 2314 019 010 CPV2L0 MR READ
MDISK 194 2314 145 058 CPV2L0 MR READ
MDISK 199 2314 034 001 CPV2L0 WR READ
```

and, optionally:

```
MDISK xxx 2314 000 203 yyyyyy MW
```

The last entry is optional and is included so that the user, MAINT, can write a CP nucleus directly to the system residence volume. If you choose to include this optional statement, replace xxx with the address of the system residence volume and yyyyyy with its volume serial number.

This directory entry is included in the Release 2 directory provided with the starter system.

THE VM/370 DIRECTORY SUPPLIED WITH THE STARTER SYSTEM

A control statement file for the VM/370 directory program is supplied with the:

- 2314 Starter System
- 3330 Starter System
- 3340 Starter System

If the supplied control statement file meets your needs, you can execute the Directory program using the supplied control statements. Otherwise, you can code your own control statements or edit the supplied control statements to produce the file you need for your installation.

2314 STARTER SYSTEM SUPPLIED DIRECTORY

The VM/370 Directory program control file supplied with the 2314 starter system is:

```
 DIRECTORY XXX 2314 LABEL
*
USER OPERATOR OPERATOR 320K 1M ABCDEG
 ACCOUNT ACT1 OPERATOR
 CONSOLE 009 3215
 SPOOL 00C 2540 READER A
 SPOOL 00D 2540 PUNCH A
 SPOOL 00E 1403 A
 MDISK 191 2314 004 010 CPV2L0 WR READ WRITE
 LINK MAINT 194 194 RR
 LINK MAINT 190 190 RR
*
USER CE CE 320K 1M EFG
 ACCOUNT ACT2 CE
 CONSOLE 009 3215
 SPOOL 00C 2540 READER A
 SPOOL 00D 2540 PUNCH A
 SPOOL 00E 1403 A
 MDISK 191 2314 014 005 CPV2L0 WR READ WRITE
 LINK MAINT 194 194 RR
 LINK MAINT 190 190 RR
*
USER MAINT CPCMS 512K 1M BCEG
 ACCOUNT ACT3 MAINT
 OPTION ECMODE REALTIMER
 CONSOLE 009 3215
 SPOOL 00C 2540 READER A
 SPOOL 00D 2540 PUNCH A
 SPOOL 00E 1403 A
 MDISK 190 2314 035 110 CPV2L0 MR READ
 MDISK 191 2314 019 010 CPV2L0 MR READ
 MDISK 194 2314 145 058 CPV2L0 MR READ
 MDISK 199 2314 034 001 CPV2L0 WR READ
*
*****
*
* MDISK XXX 2314 000 203 YYYYYY MW
*
*   THE ABOVE OPTIONAL ENTRY IS INCLUDED SO THAT THE ID: MAINT
*   CAN WRITE A CP NUCLEUS DIRECTLY TO THE SYSTEM RESIDENCE VOLUME.
*   XXX AND YYYYYY SHOULD BE CHANGED TO THE ADDRESS AND LABEL OF YOUR
*   SYSTEM RESIDENCE VOLUME AS DEFINED IN YOUR DMKSYS MODULE.
*   THE LEADING ' * ' IN FRONT OF MDISK SHOULD ALSO BE DELETED.
*
*****
USER IVPM1 IVPASS 320K 16M G
 ACCOUNT ACT4 IVPM1
 CONSOLE 009 3210
 SPOOL 00C 2540 READER A
 SPOOL 00D 2540 PUNCH A
 SPOOL 00E 1403 A
 MDISK 191 2314 001 001 CPV2L0 WR READ WRITE
 LINK MAINT 194 194 RR
 LINK MAINT 190 190 RR
```

```
*
USER IVPM2 IVPASS 320K 1M G
 ACCOUNT ACT5 IVPM2
 CONSOLE 009 3210
 SPOOL 00C 2540 READER A
 SPOOL 00D 2540 PUNCH A
 SPOOL 00E 1403 A
 MDISK 191 2314 002 001 CPV2L0 WR READ WRITE
 LINK MAINT 194 194 RR
 LINK MAINT 190 190 RR
*
USER REM2780 REMOTE 320K
 ACCOUNT ACT6 REM2780
 CONSOLE 009 3215
 SPOOL 00C 2540 READER A
 SPOOL 00D 2540 PUNCH A
 SPOOL 00E 1403 A
 SPOOL 010 2540 READER A
 SPOOL 011 2540 READER A
 MDISK 191 2314 003 001 CPV2L0 WR READ WRITE
 LINK MAINT 194 194 RR
 LINK MAINT 190 190 RR
*
USER ECMODE ECMODE 512K 1M G
 ACCOUNT ACT7 ECMODE
 OPTION ECMODE REALTIMER
 CONSOLE 009 3215
 SPOOL 00C 2540 READER A
 SPOOL 00D 2540 PUNCH A
 SPOOL 00E 1403 A
 MDISK 191 2314 029 005 CPV2L0 WR READ WRITE
 LINK MAINT 194 194 RR
 LINK MAINT 190 190 RR
*
```

3330 STARTER SYSTEM SUPPLIED DIRECTORY

The VM/370 Directory program control file supplied with the 3330 starter system is:

```
DIRECTORY XXX 3330 LABEL
*
USER OPERATOR OPERATOR 320K 1M ABCDEG
 ACCOUNT ACT1 OPERATOR
 CONSOLE 009 3215
 SPOOL 00C 2540 READER A
 SPOOL 00D 2540 PUNCH A
 SPOOL 00E 1403 A
 MDISK 191 3330 004 007 CPV2L0 WR READ WRITE
 LINK MAINT 194 194 RR
 LINK MAINT 190 190 RR
*
USER CE CE 320K 1M EFG
 ACCOUNT ACT2 CE
 CONSOLE 009 3215
 SPOOL 00C 2540 READER A
 SPOOL 00D 2540 PUNCH A
 SPOOL 00E 1403 A
 MDISK 191 3330 011 005 CPV2L0 WR READ WRITE
 LINK MAINT 194 194 RR
 LINK MAINT 190 190 RR
```

```
*
USER MAINT CPCMS 512K 1M BCEG
 ACCOUNT ACT3 MAINT
 OPTION  ECMODE REALTIMER
 CONSOLE 009 3215
 SPOOL 00C 2540 READER A
 SPOOL 00D 2540 PUNCH A
 SPOOL 00E 1403 A
 MDISK 190 3330 030 076 CPV2L0 MR READ
 MDISK 191 3330 016 007 CPV2L0 MR READ
 MDISK 194 3330 106 044 CPV2L0 MR READ
 MDISK 199 3330 029 001 CPV2L0 WR READ
*
*****
*
* MDISK XXX 3330 000 404 YYYYYY MW
*   THE ABOVE OPTIONAL ENTRY IS INCLUDED SO THAT THE ID: MAINT
*   CAN WRITE A CP NUCLEUS DIRECTLY TO THE SYSTEM RESIDENCE VOLUME.
*   XXX AND YYYYYY SHOULD BE CHANGED TO THE ADDRESS AND LABEL OF YOUR
*   SYSTEM RESIDENCE VOLUME AS DEFINED IN YOUR DMKSYS MODULE.
*   THE LEADING ' * ' IN FRONT OF MDISK SHOULD ALSO BE DELETED.
*
*****
USER IVPM1 IVPASS 320K 16M G
 ACCOUNT ACT4 IVPM1
 CONSOLE 009 3210
 SPOOL 00C 2540 READER A
 SPOOL 00D 2540 PUNCH A
 SPOOL 00E 1403 A
 MDISK 191 3330 001 001 CPV2L0 WR READ WRITE
 LINK MAINT 194 194 RR
 LINK MAINT 190 190 RR
*
USER IVPM2 IVPASS 320K 1M G
 ACCOUNT ACT5 IVPM2
 CONSOLE 009 3210
 SPOOL 00C 2540 READER A
 SPOOL 00D 2540 PUNCH A
 SPOOL 00E 1403 A
 MDISK 191 3330 002 001 CPV2L0 WR READ WRITE
 LINK MAINT 194 194 RR
 LINK MAINT 190 190 RR
*
USER REM2780 REMOTE 320K
 ACCOUNT ACT6 REM2780
 CONSOLE 009 3215
 SPOOL 00C 2540 READER A
 SPOOL 00D 2540 PUNCH A
 SPOOL 00E 1403 A
 SPOOL 010 2540 READER A
 SPOOL 011 2540 READER A
 MDISK 191 3330 003 001 CPV2L0 WR READ WRITE
 LINK MAINT 194 194 RR
 LINK MAINT 190 190 RR
```

```
*
USER ECMODE ECMODE 512K 1M G
 ACCOUNT ACT7 ECMODE
 OPTION ECMODE REALTIMER
 CONSOLE 009 3215
 SPOOL 00C 2540 READER A
 SPOOL 00D 2540 PUNCH A
 SPOOL 00E 1403 A
 MDISK 191 3330 023 006 CPV2L0 WR READ WRITE
 LINK MAINT 194 194 RR
 LINK MAINT 190 190 RR
*
*
*     CYLINDERS 150 TO 403 ARE UNUSED AND MAY BE USED FOR
*     ANY OTHER VIRTUAL MINIDISK SPACE. IT CAN ALSO BE
*     USED FOR PAGING, SPOOLING OR T-DSK SPACE.
*
*
*
```

3340 STARTER SYSTEM SUPPLIED DIRECTORY

The VM/370 Directory program control statements supplied with the 3340 starter system is:

```
DIRECTORY XXX 3340 LABEL
*
USER OPERATOR OPERATOR 320K 1M ABCDEG
 ACCOUNT ACT1 OPERATOR
 CONSOLE 009 3215
 SPOOL 00C 2540 READER A
 SPOOL 00D 2540 PUNCH A
 SPOOL 00E 1403 A
 MDISK 191 3340 007 014 CPV2L0 WR READ WRITE
 LINK MAINT 194 194 RR
 LINK MAINT 190 190 RR
*
USER CE CE 320K 1M EFG
 ACCOUNT ACT2 CE
 CONSOLE 009 3215
 SPOOL 00C 2540 READER A
 SPOOL 00D 2540 PUNCH A
 SPOOL 00E 1403 A
 MDISK 191 3340 021 006 CPV2L0 WR READ WRITE
 LINK MAINT 194 194 RR
 LINK MAINT 190 190 RR
*
USER MAINT CPCMS 512K 16M BCEG
 ACCOUNT ACT3 MAINT
 OPTION ECMODE REALTIMER
 CONSOLE 009 3215
 SPOOL 00C 2540 READER A
 SPOOL 00D 2540 PUNCH A
 SPOOL 00E 1403 A
 MDISK 190 3340 048 203 CPV2L0 MR READ
 MDISK 191 3340 027 014 CPV2L0 MR READ
 MDISK 194 3340 251 098 CPV2L0 MR READ
 MDISK 199 3340 046 002 CPV2L0 WR READ
```

```
*
*****
*
* MDISK XXX 3340 000 348 YYYYYY MW
*   THE ABOVE OPTIONAL ENTRY IS INCLUDED SO THAT THE ID: MAINT
*   CAN WRITE A CP NUCLEUS DIRECTLY TO THE SYSTEM RESIDENCE VOLUME.
*   XXX AND YYYYYY SHOULD BE CHANGED TO THE ADDRESS AND LABEL OF YOUR
*   SYSTEM RESIDENCE VOLUME AS DEFINED IN YOUR DMKSYS MODULE.
*   THE LEADING ' * ' IN FRONT OF MDISK SHOULD ALSO BE DELETED.
*
*****
USER IVPM1 IVPASS 320K 16M G
 ACCOUNT ACT4 IVPM1
 CONSOLE 009 3210
 SPOOL 00C 2540 READER A
 SPOOL 00D 2540 PUNCH A
 SPOOL 00E 1403 A
 MDISK 191 3340 001 002 CPV2L0 WR READ WRITE
 LINK MAINT 194 194 RR
 LINK MAINT 190 190 RR
*
USER IVPM2 IVPASS 320K 1M G
 ACCOUNT ACT5 IVPM2
 CONSOLE 009 3210
 SPOOL 00C 2540 READER A
 SPOOL 00D 2540 PUNCH A
 SPOOL 00E 1403 A
 MDISK 191 3340 003 002 CPV2L0 WR READ WRITE
 LINK MAINT 194 194 RR
 LINK MAINT 190 190 RR
*
USER REM2780 REMOTE 320K
 ACCOUNT ACT6 REM2780
 OPTION REALTIMER
 CONSOLE 009 3215
 SPOOL 00C 2540 READER A
 SPOOL 00D 2540 PUNCH A
 SPOOL 00E 1403 A
 SPOOL 010 2540 READER A
 SPOOL 011 2540 READER A
 MDISK 191 3340 005 002 CPV2L0 WR READ WRITE
 LINK MAINT 194 194 RR
 LINK MAINT 190 190 RR
*
USER ECMODE ECMODE 512K 1M G
 ACCOUNT ACT7 ECMODE
 OPTION ECMODE REALTIMER
 CONSOLE 009 3215
 SPOOL 00C 2540 READER A
 SPOOL 00D 2540 PUNCH A
 SPOOL 00E 1403 A
 MDISK 191 3340 041 005 CPV2L0 WR READ WRITE
 LINK MAINT 194 194 RR
 LINK MAINT 190 190 RR
*
```

ALLOCATING DASD SPACE FOR THE VM/370 DIRECTORY

Before you create your VM/370 directory using the Directory program, be sure you have enough DASD space allocated as directory space (DRCT). Use the CP Format service program to format and allocate the cylinders to be used for the VM/370 directory. The cylinders must be allocated as DRCT. To calculate the total number of cylinders required, first calculate the total number of records used:

$$NR = \frac{NU}{169} + \frac{((NU + NM) \times 2) + \text{all other control statements}}{170}$$

where:
NR = total number of records used
NU = number of USER control statements
NM = number of MDISK control statements (except for T-disks)

Then, calculate the number of cylinders NC:

- For 3330: NC = NR/57
- For 2314,2319: NC = NR/32
- For 2305,3340: NC = NR/24

Note: You should initially format and allocate space for two VM/370 directories. (See the discussion of the Format/Allocate Service Program, DMKFMT, in Appendix F.) You can then build a new directory whenever needed, without overlapping the current one, and without formatting and allocating space each time a new directory is created. If you wish to reallocate the area in which the directory resides, you must reallocate the DASD space and then rerun the Directory program. When a VM/370 directory is written, space is allocated from available cylinders, a full cylinder at a time, and a minimum of two cylinders are used for the VM/370 directory.

Once a new VM/370 directory is successfully written, cylinders used for the old directory (marked as temporarily allocated during directory creation) are marked as free. In this way, DASD space allocated for DRCT cylinders is freed and can be reused for the next directory creation. If space for two directories is not initially allocated, each time you want to create a new directory, you must allocate space for the directory before you create it.

THE DIRECTORY PROGRAM

The VM/370 Directory service program can be run under CMS (using the DIRECT command) or standalone. The standalone version of the Directory program is provided in object deck form (a three card loader, followed by the DMKDIR text deck), and may be loaded directly from either a real or virtual card reader.

If you run the Directory program under CMS, input records must be in a CMS file with a default fileid of "USER DIRECT". The DIRECT command loads the directory creation module. If no filename is specified, the program looks for a file named USER DIRECT. Otherwise, it looks for a file named filename DIRECT.

If the file is not found, or if an error occurs during processing, the directory is not created and the old directory remains unaltered.

Normal completion writes the DASD address of the new VM/370 directory in the VOL1 label, and if it is updating the active system directory, it places the new directory in use by VM/370. You can print the new directory by issuing the CMS command PRINT USER DIRECT (or PRINT filename DIRECT).

The virtual machine executing the Directory program must have write access to the volume to contain the new directory. If you create a directory that is to be written on the active VM/370 system residence volume, your virtual machine's current directory entry must have write access to the volume containing the current VM/370 directory.

Example: Assume that you have the following virtual machine for online directory modification.

```
USER OPERATNS PASSWORD 256K 1M ABC
  ACCOUNT NUMBER BIN2
    IPL CMS
    CONSOLE 009 3215
    SPOOL C 2540 READER A
    SPOOL D 2540 PUNCH A
    SPOOL E 1403 A
    LINK CMSSYS 190 190 R
    MDISK 330 3330  0 404 SYSRES WR RPASS WPASS
    MDISK 331 3330  0 404 SYSWRK WR RPASS WPASS
    MDISK 230 2314  0 203 UDISK1 RR RPASS WPASS
    MDISK 231 2314  0 203 UDISK2 RR RPASS WPASS
    MDISK 232 2314  0 203 BATCH1 RR RPASS WPASS
    MDISK 233 2314  0 203 BATCH2 RR RPASS WPASS
    MDISK 191 3330 26 010 VMDSK2 WR RPASS WPASS
```

Using the CMS EDIT command and its subcommands, you can create or modify a card-image file of the VM/370 directory input. When you are ready to write a new directory, issue the command:

```
DIRECT  filename
```

where filename is a CMS file (normally named USER) with filetype DIRECT containing the necessary Directory program control statements. The DIRECT command puts this file into the form of a directory, and replaces the old directory with this new one.

Loading the DMKDIR object deck via the card reader is the same as issuing the DIRECT command in CMS, except that after IPL, the program asks you for the address of a card reader containing the Directory program control statements.

Once the directory is updated, directory changes for a user currently logged on to the system do not take effect until the user logs off the system and then logs back on.

When a new directory is written for a new system residence volume, the new directory does not take effect until the new system residence volume is loaded (via IPL).

INVOKING THE DIRECTORY PROGRAM (DMKDIR) UNDER CMS

The VM/370 Directory service program records the configuration of each user's virtual machine in the VM/370 directory. Each virtual machine configuration includes counterparts of the components found in a real System/370: a virtual operator's console, virtual storage, and virtual I/O devices and control units.

The same version of the Directory service program deck may be placed in the card reader and loaded directly, or run in a virtual machine under CMS. For system generation, the standalone facility must be used.

The CMS file named DIRECT may be updated with the CMS Editor to include additional directory entries.

The CMS DIRECT Command

Use the CMS DIRECT command to process any file to see if it follows the required directory format. To actually change or swap the currently active VM/370 directory, you must have both of the following:

1. User class A, B, or C.
2. Write access to the system-owned (system residence or IPL device) volume that contains the current directory up to and including the directory cylinders, or to the volume that is to contain the new directory.

If you have the above qualifications and wish to verify that a CMS file can be used as a directory file, you must use the EDIT option; otherwise, if there are no control statement errors, the file is put into active use.

To build a VM/370 directory on a CP-owned volume using preallocated cylinders, a new directory should be built so as not to overlay an existing directory. You must therefore allow space for two directories, or allocate a new area for the VM/370 directory each time it is created.

If you execute the Directory Program under VM/370, the newly created directory is dynamically swapped, and placed in use by VM/370 (provided that you have class A, B or C and that the directory volume is a system-owned volume). Whether or not you have the proper privilege class the directory is updated on the directory volume. The format of the DIRECT command is:

```
┌──────────────────────────────────────────────────────────────────────┐
│           │                          ┌        ┐                      │
│  DIRECT   │  [filename  [filetype    │filemode│]]  [(EDIT)]          │
│           │                          └   *    ┘                      │
└──────────────────────────────────────────────────────────────────────┘
```

where:

filename [filetype [filemode]]
: is the identification of the file containing the control statements for the Directory Program. If no filename and filetype are given, the program defaults to a file named 'USER DIRECT;' otherwise, it looks for the file named. The filetype must be DIRECT. The filemode defaults to * if not specified.

(EDIT
: specifies that the directory is to be examined, but not changed.

Under CMS, the DIRECT command loads the directory creation module. The first statement encountered must be a DIRECTORY statement. If not found, or another DIRECTORY statement is found, the program terminates.

A syntax error in any statement generates an error message, and the directory is not updated. If no critical errors are encountered, the remaining statements are checked for syntax.

If the Directory program abnornally terminates, the old directory is not altered. Normal completion places the directory in use by VM/370. After the new directory is created, it can be printed by issuing the CMS command PRINT USER DIRECT or PRINT filename DIRECT.

The DIRECT command filename and filetype default to a CMS file identification of USER DIRECT. The filemode defaults to * if not specified. Any or all of the defaults can be overridden by the command line. The EDIT option allows you to run the program without updating the directory on disk. This enables you to check the syntax of the directory statements without accessing the directory disk.

INVOKING DIRECTORY AS A STANDALONE PROGRAM

Standalone operation is the same as CMS operation, with this exception: after IPL, the program asks you for the virtual card reader address. If you enter a null line, the IPL device address is the default of 00C.

DIRECTORY CONTROL STATEMENTS

The control statements should be in the following formats, with one or more blanks as operand delimiters. All operands are positional from left to right. If any operands are omitted, all remaining operands in that statement must be omitted, with the exception of the OPTION statement. Its entries are self-defining and not positional.

Only columns 1 through 71 are inspected by the program. All data after the last possible operand on any card is ignored. Also, blank cards and cards having an asterisk (*) as the first operand are ignored.

If any input card is found to be in error, the program continues to process the control statements, validating all control statements before

terminating. If the directory runs out of space, the program terminates immediately. After an abnormal termination (or, for CMS, the EDIT run), the old directory is not altered, and the new directory is not saved.

DIRECTORY Control Statement

The DIRECTORY control statement defines the device on which the directory is allocated. It must be the first statement. The format of the DIRECTORY control statement is:

```
DIRectory  cuu  devtype  volser
```

where:

cuu is the input device address, specified in 3 hexadecimal digits.

devtype is four decimal digits that represent a supported device type suitable for the VM/370 directory (2314, 2319, 2305, 3330, or 3340).

volser is the volume serial number of the directory volume (one to six alphameric characters).

USER Control Statement

The USER control statement defines a virtual machine and creates a VM/370 directory entry. It delimits the directory entry for one user. A separate USER statement must be prepared for each directory entry required. The format of the USER control statement is:

```
                                          ┌    ┌    ┌    ┌    ┐┐┐┐
User userid pass [stor [mstor [cl [pri    |le  |ld  |cd  |es  ||||]]]]
                                          |ON  |ON  |ON  |ON  ||||
                                          |OFF |OFF |OFF |OFF ||||
                                          └    └    └    └    ┘┘┘┘
```

where:

userid is a one- to eight-character user identification. Any alphameric characters may be used except SYSTEM. SYSTEM is the userid of the VM/370 system VMBLOK, and should never be used for a virtual machine. Each user in the directory, except for those whose password is NOLOG, must have at least one device. Any of the various devices described meet this requirement; for example, the device may be a console or spool device.

Notes:

1. The userid should not contain the characters "LOGONxxx", where xxx is a terminal address of the installation. This character string is assigned to the terminal at address xxx from the time the initial interrupt is received until the user is identified, during log on.

2. Do not specify SYSTEM as a userid. VM/370 reserves SYSTEM as an identifier for its own use.

pass is a one- to eight-character user security password that must be entered by the user to gain access to the VM/370 system and the virtual machine you are defining in these control statements.

Note: Do not use the password NOLOG as it is reserved for users who do not have a virtual machine configuration in the VM/370 directory. NOLOG is used for spooling purposes only; attempts to log on using this password are inhibited.

stor is one to eight decimal digits that define the virtual machine's storage size. It must be a multiple of 4K. The last character must be K or M. The default is 128K. The minimum size is 8K. All entries not on a 4K boundary are rounded up to the next 4K boundary. The maximum size is 16M.

mstor is one to eight decimal digits that define the maximum virtual machine storage size that this user can define as his storage after logging on the system. It must be coded in multiples of 4K. The last digit must be K or M. The default size is 1M. All entries not on a 4K boundary are rounded to the next 4K boundary. The minimum size is 8K. The maximum size that can be specified is 16M.

cl is one to eight characters from A to H (with no intervening blanks) defining the privilege class or classes given to this user. The default is G.

Note: If privilege class F is assigned to a virtual machine, I/O error recording is not automatically done. This allows the class F user to set the kind of error recording he wants to perform.

pri is a number from 1 to 99 used by the control program priority dispatcher. One is the highest priority and fifty is the default.

Note: The same priority value can be used for several users. Also, if the specification for this statement is not entered, then line end (le), line delete (ld), character delete (cd), and escape (es) characters default to system-defined values.

The following special VM/370 logical editing symbols may be set ON, OFF, or substituted with two hexadecimal characters or one graphic character of the user's choice.

Note: In addition to the directory specification, the user can change these logical editing symbols using the TERMINAL command. The default value for all symbols is ON.

le is a one-character "line end" symbol or a two-character hexadecimal representation of the symbol. ON sets the system default value (#). OFF disallows "line end" symbol usage. For example:

"le" can be coded as + or 4D or ON or OFF.

ld is a one-character "line delete" symbol or a two-character hexadecimal representation of the symbol. ON sets the system default (¢). OFF disallows "line delete" usage.

cd is a one-character "character delete" symbol or a two-character hexadecimal representation of the symbol. ON sets the system default value (@). OFF disallows "character delete" usage.

es is a one-character "escape-character" symbol or a two-character hexadecimal representation of the symbol. ON sets the system default ("). OFF disallows "escape character" symbol usage.

ACCOUNT Control Statement

The ACCOUNT control statement defines an account number and a distribution identification. The distribution identification has no internal system use; it is provided for customer use (for example, a code for distribution of printed output). The ACCOUNT statement is optional. This statement (if coded) must follow the USER statement and precede the first device statement. The format of the ACCOUNT control statement is:

```
Account number [distribution]
```

where:

number is a one- to eight-character account number that is punched in the accounting data for this virtual machine. The USERID from the USER statement is also punched in the accounting data.

distribution is a one- to eight-character distribution identification word that is printed or punched with the userid in the separator for spooled output for this user. This value is optional and defaults to the userid from the USER statement if omitted.

OPTION Control Statement

The OPTION control statement selects specific options available to the user. This statement is optional and, if used, must follow the USER statement and precede the first device statement (CONSOLE, MDISK, DEDICATE, LINK, or SPOOL). The format of the OPTION control statement is:

```
Option Realtimer Ecmode Isam Virt=real Acct Svcoff BMX
```

where:

REALTIMER provides a timer for the virtual machine that is updated during virtual CPU run time and also during virtual wait time. (If the virtual machine does not have the REALTIMER option, its timer reflects only the virtual CPU

run time used.) This option is required for RSCS and for virtual machines running systems or programs that go into a wait state expecting a timer interruption. This timing ability can also be obtained by issuing the CP command line SET TIMER REAL.

ECMODE allows the virtual machine to run in extended control mode. The ECMODE option must be specified for virtual machines using operating systems that:

1. Operate in System/370 extended control mode (such as VM/370 itself).
2. Use the dynamic address translation facility (such as OS/VS1, OS/VS2 and VM/370).
3. Use extended control registers other than zero (such as OS GTF [General Trace Facility], which uses Monitor Call and requires extended control register eight).
4. Depend on the System/370 extended channel masking feature.

The ECMODE option must also be specified for the virtual machine that is to perform system support or updating.

A virtual machine defined without the ECMODE option in the directory is limited to six I/O channels, while a virtual machine with the ECMODE option may address up to 16 I/O channels. If a virtual machine with the ECMODE option executes in basic control mode, the I/O masking for channels 6 and higher is simulated by the extended channel feature. If a virtual machine with the ECMODE option executes in extended control mode, the I/O masking for all sixteen channels is handled via extended control register 2.

This facility can also be obtained by issuing the CP command line SET ECMODE ON.

ISAM provides special channel command word translation routines that permit OS/PCP, MFT, and MVT ISAM programs (which dynamically modify their CCWs) to operate properly in a virtual machine. This is required only for virtual machines that use OS/PCP, MFT, or MVT ISAM access methods or VS1 ISAM when executing in a V=R partition under VS1. This option is not needed for DOS, DOS/VS, or OSVS1 ISAM when run only in a V=V partition of VS1.

This facility can also be obtained by issuing the CP command line SET ISAM ON.

VIRT=REAL is a performance option that allows the user to place his virtual machine in lower storage, such that its virtual storage addresses correspond to the real storage address (except for its page zero, which is relocated). The real page zero is controlled by the CP nucleus. No CCW translation is required. This option is required for a virtual machine to successfully execute self-modifying channel programs other than those generated by OS/VS TCAM (Level 5, generated or invoked with the VM/370 option) or OS ISAM. VIRT=REAL can be specified for any number of

virtual machines but only one virtual machine can use this facility at any given time.

To generate a VM/370 system with a virtual=real machine, see the "Specifying a Virtual=Real Machine" section of Part 1.

| ACCT | a user with the ACCT option in his directory can charge another user for virtual machine resources. For example, a user who sends a job to the CMS batch virtual machine can be charged for the time that he uses in the batch machine.

| SVCOFF | specifies that CP, instead of the Virtual Machine Assist feature, handles all SVC interrupts for this virtual machine. A user whose directory entry contains this option can override it by issuing SET ASSIST SVC.

| | Note: All SVC 76 interrupts are handled by CP whether or not the SVCOFF option is specified.

| BMX | specifies that all virtual machine I/O operations are to occur as block multiplexer channel operations rather than selector channel (the default) operations. In block multiplexer mode, the virtual channel is not busy until the initial SIO is complete (selector mode operates similarly). Block multiplexer allows the successful start of multiple SIOs to different devices on the same channel. However, virtual I/O operations on channel 0 are processed as byte multiplexer channel operations. Channels that have a channel-to-channel adapter are restricted to selector channel operation.

| | Note: The channel mode setting can be changed by use of the CP DEFINE command.

IPL Control Statement

The IPL control statement contains a one- to eight-character name of the system (or one- to three-digit I/O device address) to be loaded for the user when he logs on. This statement is optional; if specified, it must follow the USER statement, and must precede the first device statement (CONSOLE, MDISK, or SPOOL). The IPL statement can be overridden by the user at logon time by specifying "LOGON userid NOIPL". (Note: If the user is the primary system operator, an automatic IPL is not performed when he logs on.) The format of the IPL statement is:

```
Ipl  iplsys
```

where:

| iplsys | is a one- to eight-character system name or the virtual address of the device containing the system to be loaded.

CONSOLE Control Statement

The CONSOLE control statement specifies the virtual console. The format of the CONSOLE control statement is:

```
Console  cuu  devtype [class]
```

where:

cuu is the virtual device address of one to three hexadecimal digits.

devtype is the device type:
- 1052
- 3210
- 3215

Note: The system accepts any of the devtypes indicated regardless of the real console or terminal being used. Device types 3277, 3066, 3158, 2741, and 3767 cannot be specified. Only one console can be specified. If a different console is sometimes required, use the CP DEFINE command to change the console address or add an alternate console.

class is a one character spooling class. A through Z and 0 through 9 are valid. The class governs the printing of the real spooled output. If the class operand is omitted, the default is class T and is for console spooling.

MDISK Control Statement

The MDISK control statement describes the cylinder extent on a direct access device to be owned by the user. The DASD area assigned with this statement becomes the user's minidisk. The format of the MDISK control statement is:

```
Mdisk cuu devtype {cylr    cyls volser [mode [pr [pw [pm]]]]}
                  {T-disk  cyls                              }
```

where:

cuu is the virtual device address of one to three hexadecimal digits.

devtype is the device type:
- 2305
- 2311 Top (Top half of a 2314 or 2319)
- 2311 Bottom (Bottom half of a 2314 or 2319)
- 2314
- 2319
- 3330
- 3340

{cylr / T-disk} is a three-digit decimal cylinder relocation factor which specifies the cylinder on a real disk that corresponds to cylinder 0 of the virtual disk. If T-disk is specified,

temporary disk space is obtained at logon time from preallocated system disk space. This space must be initialized or formatted by the user when he logs on and is a part of his virtual configuration until he logs off or detaches the disk, at which time the data area is returned for reallocation for another T-DISK area. To maintain security this area should be physically erased before it is returned.

Note: It is not advisable to define that a minidisk start at real cylinder zero (unless the minidisk is to be used by OS ISAM, in which case it must begin at real cylinder zero). If you do assign a minidisk beginning at real cylinder zero, the user who owns it must realize that the minidisk label is the real label that both he and the VM/370 system use to identify the disk. Also note that CP-owned volumes must not have minidisks beginning at real cylinder zero.

cyls is a one- to three-digit decimal number specifying the number of cylinders. The maximum number of cylinders that can be specified for a minidisk is 203 cylinders on a 2314, 404 cylinders on a 3330 Model 1 or 2, 808 cylinders on a 3330 Model 11, 349 cylinders on a 3340 with data module 35, and 698 cylinders on a 3340 with Data Module 70. The maximum number of cylinders supported by CMS for a single minidisk is 246 on a 3330 device and 682 on a 3340 device.

volser is the volume serial number of the DASD volume (one to six alphameric characters).

mode is the primary access mode requested for the device (read-only, write, or multiple-write), and the alternate access (read-only or write) desired (if any), as follows:

- R specifies that read-only (R/O) access is requested. The access is not given if any other user has the disk in write status.
- RR specifies that read-only access is requested, even if another user has the disk in write status.
- W specifies that write access is requested. The disk is not defined if any other user has the disk in read or write status.
- WR specifies that write access is requested if no other user has the disk in read or write status, but that an alternate access of read-only is acceptable if others do have a link to the disk.
- M specifies that multiple access is requested. This means that a write link is to be given to the disk unless another user already has write access to it, in which case no link is to be given.
- MR specifies that a write link is to be given to the disk unless another user already has write access to it. In this case, a read link is given to the user and the 'DEV xxxx FORCED R/O' message is issued.
- MW specifies that a write link is to be given to the disk in any case.

If a mode specification is omitted from the statement, it defaults to W.

pr is the password that allows sharing in read mode (a one- to eight-character field).

pw is the password that allows sharing in write mode (a one- to eight-character field).

pm is the password that allows sharing in multiple-write mode (a one- to eight-character field).

A write password (pw) cannot be specified without a read password (pr); a multiple-write password (pm) cannot be specified without both a read password (pr) and a write password (pw).

Note: If a password of ALL is used for pr, pw, or pm, a user other than the owner of the minidisk is allowed to link to this minidisk without specifying a password.

Examples:

MDISK 230 3330 5 10 WORK01 W ALL WRITE

is an MDISK statement for a minidisk with read/write access to 10 cylinders located on a real 3330 disk volume labeled WORK01, beginning at real cylinder 5. A user other than the owner of this minidisk can link to it in read status without specifying a read password, but must specify a password of 'WRITE' in order to gain write access to it.

MDISK 191 2314 50 15 CPDSK4 W RDPASS WRX2*

is an MDISK statement for a minidisk with read/write access to 15 cylinders located on a real 2314 labeled CPDSK4 starting at cylinder 50. A read password of RDPASS and a write password of WRX2* are provided. This allows the other users to access the minidisk through the directory LINK statement (see the description of the LINK statement in this section) or the LINK command.

SPOOL Control Statement

The SPOOL control statement specifies the unit record device that is to be spooled. Multiple readers, punches, and printers may be specified, each on a separate SPOOL card. The format of the SPOOL control statement is:

```
Spool  cuu devtype  [class]
```

where:

cuu is the virtual device address (one to three hexadecimal digits).

devtype is the device type:
1403
2501
1443
3211
2540 R[EADER]
2540 P[UNCH]
3525
3505

class is a one-character spooling class. A through Z, 0 through 9, or * is valid. For spool output devices, the class governs the punching or printing of the real spooled output. If this operand is omitted, the default class A is used. This operand is required for all output devices defined on the spool record. For spool input devices, the class controls access to spool files by virtual card readers. The default class for readers is an asterisk (*), which means the reader can process any class of spool file.

For example:

SPOOL 00E 1403 A

specifies a SPOOL record for a virtual 1403 at address 00E. The output class is A.

DEDICATE Control Statement

The DEDICATE control statement specifies that a real device is to be dedicated to this user. If the device is a unit record device, input and output are not spooled by VM/370. A real device may be dedicated to only one user at a time. Should a device be specified as dedicated in more than one directory entry, only the first user to log on gains access to it. The format of the DEDICATE control statement is:

```
| Dedicate  cuu { rdev |[VOLID] volser } [R/O]
```

where:

cuu is the virtual device address (one to three hexadecimal digits).

rdev is the real device address (one to three hexadecimal digits).

VOLID volser
is an optional keyword used if the volser is less than four characters long. It is optional if volser is four, five, or six characters long.

The variable, volser, is the volume serial number of a disk pack mounted on some real disk storage device (one to six alphameric characters).

If the VOLID operand is used, the volume must be attached to the system when the user logs on. When he logs off, the operator can then detach the volume from the system.

| R/O specifies that the virtual device is to be in read-only status. If this operand is omitted, the status defaults to read/write.

Examples:

DEDICATE 0B8 0B0

is a DEDICATE statement for a device at real address 0B0. Its virtual address is 0B8.

DEDICATE 250 MYPACK

is a DEDICATE statement that defines, for this virtual machine, virtual address 250 as the real device where DASD volume MYPACK is mounted.

LINK Control Statement

The LINK control statement makes a device that belongs to another user (userid) available to this virtual machine at logon time. If you want to make one volume available to several virtual machines:

- Define the volume for one of the virtual machines with an MDISK statement, and
- Define a link to that volume, with the LINK statement for all other virtual machines that use the volume.

Then, if you later must move or change that volume you only need to update the one MDISK statement, the LINK statements need not be updated. The format of the LINK control statement is:

```
Link  userid  ldev  [cuu [mode]]
```

where:

userid — is the one- to eight-character user identification of the user to be linked to.

ldev — is the virtual device address of the device owned by userid to be linked to (three hexadecimal digits). This is the virtual device address, assigned by userid, of the disk you wish to link to.

cuu — is the virtual device address for the virtual machine being defined. "cuu" defaults to the same address as the linked-to device (three hexadecimal digits). If your virtual machine has the ECMODE option, any address up to X'FFF' is valid; otherwise, any address up to X'5FF' is valid.

mode — is the primary access mode requested for the device (read-only, write, or multiple-write), and the alternate access (read-only or write) desired, if any, as follows:

- R — specifies that read-only (R/O) access is requested. The link is not given if any other user has the disk in write status.
- RR — specifies that read-only access is requested, even if another user has the disk in write status.

W specifies that write access is requested. The disk is not defined if any other user has the disk in read or write status.

WR specifies that write access is requested if no other user has the disk in read or write status, but that an alternate access of read-only is acceptable if others do have a link to the disk.

M specifies that multiple access is requested. This means that a write link is to be given to the disk unless another user already has write access to it, in which case no link is to be given.

MR specifies that a write link is to be given to the disk unless another user already has write access to it. In this case, a read link is to be given to the user, and the 'DEV xxxx FORCED R/O' message is issued.

MW specifies that a write link is to be given to the disk in any case.

Note: If the mode is not specified, the default is R.

It is the responsibility of the operating system executing in each virtual machine to keep data from being destroyed or altered on shared disks.

CMS supports multiply-accessed read-only disks in full. CMS does not support write access to disks by multiple users. A disk accessed in write mode by one CMS user is available to other CMS users in read-only mode, but those files altered by the write-mode user cannot be read by the other users.

CMS disks should never be linked in write mode to more than one user. If two or more CMS users have write access to the same disk, all data on the disk may be destroyed.

SPECIAL Control Statement

The SPECIAL control statement specifies the I/O units available to the user that need not have a real I/O unit available. Special devices are program simulated devices that may or may not be connected to real or virtual devices after the user has logged off. The format of the SPECIAL control statement is:

```
SPEcial  cuu  devtype  [IBM|Tele]
```

Directory Program

where:

cuu is a one- to three-character virtual device address.

devtype is the device type:

2701
2702
2703
3158 (virtual 3158 console)
3270 (virtual 3270 only)
CTCA (channel to channel adapter)
(The virtual address must be a control unit address -- last four bits set to zero -- resulting in 16 device addresses being defined.)

TIMER (pseduo-timer device)

IBM
TELE
valid only if devtype is 2701, 2702, or 2703

For example, a virtual machine executing a multiple-access system that supports four IBM Type 1 adapter lines, would have four SPECIAL entries, one for each of those addresses. This provides a virtual 270X line to allow a user to dial to this multiple-access system rather than logging on as a separate virtual machine.

Notes:

1. The Integrated Communications Attachment (ICA) on the System/370 Model 135 should be specified as a 2701.

2. The 3270 is only supported by VM/370 as a locally attached device. It is not supported as a remote terminal on leased or switched lines.

The system name table consists of entries that identify the name and location of saved systems. Two macros generate entries for the system name table:

- The NAMESYS macro creates an entry in the system name table for a virtual machine operating system.
- The NAMENCP macro creates an entry in the system name table for a 3704/3705 control program.

A system name table is supplied for each starter system. The DMKSNT module supplied with the 2314 starter system is:

```
DMKSNTBL CSECT
CMS      NAMESYS SYSSIZE=256K,SYSNAME=CMS,                              X
                 VSYSADR=190,SYSVOL=VMREL2,SYSCYL=035,SYSSTRT=(001,1), X
                 SYSPGCT=35,SYSPGNM=(0-34),SYSHRSG=1,VSYSRES=CPV2L0
         END
```

The DMKSNT module supplied for the 3330 starter system is:

```
DMKSNTBL CSECT
CMS      NAMESYS SYSSIZE=256K,SYSNAME=CMS,                              X
                 VSYSADR=190,SYSVOL=VMREL2,SYSCYL=030,SYSSTRT=(001,1), X
                 SYSPGCT=35,SYSPGNM=(0-34),SYSHRSG=1,VSYSRES=CPV2L0
         END
```

The DMKSNT module supplied for the 3340 starter system is:

```
DMKSNTBL CSECT
CMS      NAMESYS SYSSIZE=256K,SYSNAME=CMS,                              X
                 VSYSADR=190,SYSVOL=VMREL2,SYSCYL=048,SYSSTRT=(001,1),  X
                 SYSPGCT=35,SYSPGNM=(0-34),SYSHRSG=1,VSYSRES=CPV2L0
         END
```

The supplied DMKSNT modules each have only one entry: an entry that can save a copy of CMS if you use all the recommended labels and allocations and the starter system supplied DMKSYS when you follow the system generation procedure. If you wish to change or add to the system name table that is supplied, you must code your own macro and create a DMKSNT file of your own. The system generation procedure tells you when to assemble your own file. If the supplied DMKSNT module meets your installation's needs, you do not have to code or assemble the DMKSNT module.

If you do create your own version of the system name table, your file must have a CSECT and END statement:

```
DMKSNTBL CSECT
         NAMESYS macros (one for each virtual machine operating
                 system you wish to save)
         NAMENCP macros (one for each 3704/3705 control program
                 you create)
         END
```

Information about coding the NAMESYS and NAMENCP macros follows.

Note that the supplied module contains a PUNCH SPB (Set Page Boundary) card, which is used by the loader to force this module to a 4K boundary when the CP system is built. A 12-2-9 multipunch must be specified in column 1 of an SPB card. Be sure that the SPB card is not deleted.

CODING THE NAMESYS MACRO

The NAMESYS macro describes the location of the saved system. Shared segments may be specified, but they must consist of reenterable code, with no alteration of its storage space permitted.

The format of the NAMESYS macro is:

```
| label | NAMESYS | SYSSIZE=nnnK,SYSNAME=name,VSYSRES=cccccc,          |
|       |         | VSYSADR=cuu,SYSVOL=cccccc,SYSCYL=nnn,              |
|       |         | SYSSTRT=(cc,p),SYSPGCT=pp,                         |
|       |         | SYSPGNM=(nn,nn,nn-nn,...),                         |
|       |         | SYSHRSG=(s,s,...)                                  |
```

where:

label is any desired user label.

SYSSIZE=nnnK
: is up to three decimal digits representing the amount of storage you must have available in order to IPL the saved system. K must be specified.

SYSNAME=name
: is the name (up to six alphameric characters) given to the system to be used for identification by the SAVESYS and IPL commands. The name selected must never be one that could be interpreted as a hexadecimal device address (for example, 'A' or 'E').

VSYSRES=cccccc
: is the real volume serial number (up to six alphameric characters) of the DASD volume containing the minidisk that is the system residence volume of the system to be saved.

VSYSADR=cuu
: is the virtual address of the minidisk that is the system residence volume of the system to be saved.

SYSVOL=cccccc
: is the volume serial number (up to six alphameric characters) of the DASD volume designated to receive the saved system. This must be a CP-owned volume.

SYSCYL=nnn
: is the real starting cylinder of the minidisk (specified by VSYSRES and VSYSADR) that is the system residence volume of the system to be saved.

SYSSTRT=(cc,p)
: designates the starting cylinder (cc) and page address (p) on SYSVOL at which this named system is to be saved. During the SAVESYS and IPL command processing, this is used to generate the "cylinder page and device" address for the DASD operations. These numbers are specified in decimal.

: The number of pages written to this area is the total number specified via the SYSPGNM operand, plus one information page.

SYSPGCT=pp
: is the total number of pages (pp) you specify to be saved (that is, the total number of pages you indicate via the SYSPGNM operand). This is a decimal number, up to two digits.

SYSPGNM=(nn,nn,nn-nn,...)
: are the numbers of the pages to be saved. Pages may be specified singly or in groups. For example: if pages 0, 4, and 10 through 13 are to be saved, use the format: SYSPGNM=(0,4,10-13).

SYSHRSG=(s,s,...)
: are the segment numbers designated as shared. The pages in these segments are set up at IPL time to be used by any user loading by this name. All segments to be shared must be reentrant.

For example, a DMKSNT module to create a named CMS system could be coded as follows:

```
DMKSNTBL CSECT
FSTNAME NAMESYS  SYSSIZE=320K,SYSNAME=CMS,VSYSRES=CPDSK1,            X
                 VSYSADR=190,SYSCYL=100,SYSVOL=CPDSK2,               X
                 SYSSTRT=(400,1),SYSPGCT=33,                         X
                 SYSPGNM=(0-32),SYSHRSG=(1)
        END
```

Information on the following subjects is in the VM/370: System Programmer's Guide.

- Determining when to save a system.
- Using the SAVESYS command.
- Saving the CMS system.
- Saved system restrictions for CMS.
- Saving OS.

CODING THE NAMENCP MACRO

You must create an entry in the system name table (DMKSNT) for each unique 3704/3705 control program that you generate. If you can foresee generating several versions of the 3704/3705 control program, define extra entries in the system name table when you generate VM/370. In this way, you do not have to regenerate the VM/370 system just to update the system name table.

Use the NAMENCP macro to define 3704/3705 program entries in the system name table. The format of the NAMENCP macro is:

```
label | NAMENCP | CPSIZE=nnnK,
      |         | CPNAME=ncpname,
      |         | CPTYPE={EP  }
      |         |        {NCP }
      |         |        {PEP },
      |         | SYSPGCT=pp,
      |         | SYSVOL=volser,
      |         | SYSSTRT=(ccc,p)
```

where:

label is any desired user label.

CPSIZE=nnnK is the storage size of the 3704/3705 specified during the 3704/3705 control program generation.

CPNAME=ncpname is the name of the 3704/3705 control program image. This name is used in the SAVENCP and NETWORK LOAD commands. The name must be from one to eight alphameric characters.

CPTYPE={EP } is the 3704/3705 control program type.
{NCP }
{PEP }

SYSPGCT=pp is the total number of pages (pp) to be saved. This value may be equal to the number of pages implied by the CPSIZE operand plus four pages, but it must not exceed that total.

SYSVOL=volser is the volume serial number (volser) of the DASD volume designated to receive the control program image. That volume must be a CP-owned volume.

SYSSTRT=(ccc,p) is the starting cylinder (ccc) and page address (p) on SYSVOL at which this image is to be saved. These numbers must be specified in decimal.

Altering the Forms Control Buffer Load (DMKFCB)

The DMKFCB module is supplied with the starter system. This module defines a 3211 forms control buffer image with 6 lines/inch, 66 lines/page and the following channel skip specifications:

Line Represented	Channel Skip Specification
1	1
3	2
5	3
7	4
9	5
11	6
13	7
15	8
19	10
21	11
23	12
64	9

If you wish to alter the supplied buffer load, see the VM/370: System Programmer's Guide for directions.

Part 3: Generating VM/370 (CP, CMS, and RSCS)

Part 3 describes the step-by-step generation procedures for CP, CMS, and RSCS and describes the Installation Verification Procedure (IVP) for CP and CMS. It contains the following sections:

- Generating CP and CMS Using the 2314 Starter System
- Generating CP and CMS Using the 3330 Starter System
- Generating CP and CMS Using the 3340 Starter System
- Verifying CP and CMS Using the IVP
- Generating and Installing RSCS

Before you start to install VM/370, be sure you have the following available:

- Two real disk drives
- At least one real tape drive (the system generation process is simpler if you use two tape drives)
- Two scratch disks (one is used for the starter system and the other is used for the new system residence volume)
- The starter system tape(s)
- The PLC tape
- At least one scratch tape

The system generation procedures described in this manual refer to related publications. These publications are listed in the Preface.

The following procedures for installing CP and CMS are described:

- 2314 starter system
- 3330 starter system
- 3340 starter system

Following the procedures for installing CP and CMS are the procedures for installing the optional RSCS (Remote Spooling Communications Subsystem).

In addition, the information you need to install the 3704/3705 control program is found in "Part 4: Generating the 3704/3705 Control Program" and the information you need to install the standalone 2780 Spool Remote Program is in "Part 5: The Standalone Spool Remote Program."

GENERAL INFORMATION

CP and CMS have separate system residence disks which may be located on the same or different physical disks. The following procedure tells you how to generate the CP system residence disk or move the CMS system residence disk. Before you attempt to generate a VM/370 system, make sure that the real I/O configuration file (DMKRIO), CP system control file (DMKSYS), the VM/370 Directory file (DMKDIR), and, optionally, the forms control buffer load (DMKFCB) and the system name table file (DMKSNT) are punched. Information about preparing these files is in "Part 2: Defining Your VM/370 System." If CMS is to be saved as a named system, be sure that the NAMESYS macro is coded correctly in the DMKSNT file.

The VM/370 system is distributed on four 9-track tapes (1600 or 6250 bpi), or six 9-track tapes (800 bpi) that can be restored to direct access volumes. You must specify the device type (2314/2319, 3330 or 3340) when you order VM/370. You should also specify the tape density required (800, 1600, or 6250 bpi).

These basic tapes are as follows:

- The VM/370 starter system (one or two tapes) contains the base level of both the CP and CMS systems, and the text decks with which to build these systems.
- The CP2 tape contains all source files, text files, support procedures, macros, and the macro library of CP.
- The CMS2 file (one or two tapes) contains all source files, text files, module files, support procedures, macros, and the macro library of CMS.
- The PLC tape contains all source updates, text decks, modules, macros and macro libraries, and procedures required to build the latest level of CP, CMS, and RSCS.

Three optional sets of tapes may also be ordered:

- The Assembler tape containing source, macros, text, modules, and procedures for the Assembler.
- CP assembly listings (three tapes).
- CMS assembly listings (two tapes).

Generating CP and CMS Using the 2314 Starter System

Except where otherwise noted, you can substitute other values in place of the device addresses, volume labels, and allocations shown. Note that if you use the sample DMKSNT and DMKSYS provided with the starter system, and use the sample allocations shown in Steps 2 and 3, you can save your CMS system at the end of the procedure.

It is strongly recommended that you use the sample allocations given in Step 2 and the label VMREL2 for the new system residence volume, to ensure that you have sufficient TEMP space to complete the system generation. (The TEMP space provided on the starter system volume may not be sufficient for large systems.)

| Note: The 2314 starter system cannot be generated on a real System/370 | with a console address greater than 5FF hexadecimal.

The examples of messages and responses assume that you are performing the system generation at a typewriter terminal, such as a 3210, 2741, or 3767 (operating as a 2741). If you are using a display device, such as the 3277, when you type the response to a prompting message, that response appears in the user input area. When you enter that response, it is redisplayed in the output area on the line below the prompting message. Also, if the standalone service programs (such as the DASD Dump Restore program or Format/Allocate program) send output to a terminal display screen, the output is wrapped around immediately, when the screen becomes full, to continue displaying.

While you are generating the system, you may see some extraneous messages as the starter system is processing. These are not shown in the examples below. Only those messages that you should take note of or respond to are shown.

STEP 1. LOAD THE FORMAT PROGRAM FROM THE STARTER SYSTEM TAPE

Mount the CP starter system tape and IPL the tape. The CP Format/Allocate service program is the first file on the tape; it is now loaded. Do not rewind the tape because the next file is needed later in the system generation procedure (Step 4).

STEP 2. FORMAT, LABEL, AND ALLOCATE THE SYSTEM RESIDENCE VOLUME

| Use the CP Format/Allocate program to format, label, and allocate space on the new system residence volume. First, identify the system console by pressing the Request key (or equivalent); if the console address is either 009 or 01F, you do not have to press the Request key. Then, to | execute the Format/Allocate service program, respond to the prompting messages.

In the following example, the responses (format, 131, 2314, 000, 202, and vmrel2) format the 2314 disk at address 131 and label it VMREL2. Whatever label you specify must match what you specified on the SYSVOL

operand of the SYSRES macro when you defined the system. In any case, do not use CPV2L0 because that is the label of the starter system disk. The console output is:

```
VM/370 FORMAT/ALLOCATE PROGRAM VERSION 2.0

ENTER FORMAT OR ALLOCATE:format
FORMAT FUNCTION SELECTED
ENTER DEVICE ADDRESS (CCU):131
ENTER DEVICE TYPE:2314
ENTER START CYLINDER (XXX) OR "LABEL":000
ENTER END CYLINDER (XXX):202
ENTER DEVICE LABEL:vmrel2
FORMAT STARTED
FORMAT DONE
000 PAGE RECORDS FLAGGED
```

When the format operation completes, the prompting message

```
ENTER FORMAT OR ALLOCATE:
```

is displayed. Now that the system residence volume is formatted and labeled, you must allocate the disk space. Again, you must respond to the prompting messages. In the following example, the space on the 2314 disk at address 131, with the label VMREL2, is allocated (9 cylinders of permanent space, 4 cylinders of directory, 177 cylinders of temporary space, and 13 cylinders of TDSK space.) You can use the formulas given in the "Creating Your VM/370 Directory" section of Part 2 to ensure that 4 cylinders is enough space for your VM/370 directory. If you do not allocate your DASD space as shown in this example, you are responsible for ensuring that you have enough temporary space to perform the assemblies associated with VM/370 system generation.

```
ENTER FORMAT OR ALLOCATE:allocate
ALLOCATE FUNCTION SELECTED
ENTER DEVICE ADDRESS(CCU):131
ENTER DEVICE TYPE:2314
ENTER DEVICE LABEL:vmrel2
ENTER ALLOCATION DATA FOR VOLUME VMREL2
TYPE CYL CYL
.... ... ...
perm 000 008
drct 009 012
temp 013 189
tdsk 190 202
end
ALLOCATION RESULTS
PERM 000 008
DRCT 009 012
TEMP 013 189
TDSK 190 202
DEVICE 131 VOLUME VMREL2 ALLOCATION ENDED
ENTER FORMAT OR ALLOCATE:
```

STEP 3. LABEL THE STARTER SYSTEM VOLUME

Use the Format/Allocate program to label the scratch volume that is to contain the CP starter system. This label must be CPV2L0. You can format and label this volume, or just label it. Formatting is unnecessary because you are going to restore the starter system to this volume. In the following example, the responses (format, 130, 2314,

label, and cpv2l0) to the prompting messages put the label CPV2L0 on the 2314 disk at address 130.

```
ENTER FORMAT OR ALLOCATE: format
FORMAT FUNCTION SELECTED
ENTER DEVICE ADDRESS (CCU):130
ENTER DEVICE TYPE:2314
ENTER START CYLINDER (XXX) OR "LABEL":label
ENTER DEVICE LABEL: cpv2l0
LABEL IS NOW CPV2L0
```

When the Format/Allocate program is complete, it responds:

```
ENTER FORMAT OR ALLOCATE:
```

You do not have to respond to this message.

Now the volume is available and ready for the data that is to be placed on it by the DASD Dump Restore service program (module DMKDDR). The DASD Dump Restore program is the second file on the starter system tape.

STEP 4. LOAD THE DASD DUMP RESTORE PROGRAM FROM THE STARTER SYSTEM TAPE

IPL the starter system tape a second time to load the DASD Dump Restore (DDR) program. It is the second file on the starter system tape. Do not rewind the tape, because the next file is needed in Step 5.

STEP 5. RESTORE THE STARTER SYSTEM TO DISK

Respond to the DDR prompting messages to restore the starter system.

In the following example, the starter system is restored from the 2400 series tape drive at address 280 to the 2314 disk at address 130 (and with label CPV2L0.) The console output is:

```
VM/370 DASD DUMP/RESTORE PROGRAM VERSION 2.0
ENTER CARD READER ADDRESS OR CONTROL STATEMENTS
ENTER: sysprint cuu (cuu=real printer address)
ENTER: input 280 2400 (See the "Special Considerations for 800 BPI Tapes"
       section, if you ordered your starter system on 800 bpi tape)
ENTER: output 130 2314 cpv2l0
ENTER: restore all
RESTORING CPV2L0
END OF RESTORE
ENTER: (null line -- END key on 3215 or Enter key on 3277)
END OF JOB
```

The DDR program restores the third file on the starter system tape to the disk labeled CPV2L0. The restored disk contains:

- A VM/370 directory file (RELEASE2 DIRECT), as well as sample source for DMKSYS, DMKSNT, and DMKFCB
- A VM/370 starter system nucleus
- A CMS system residence volume
- A CP system comprised of macro libraries and text files

When the disk is restored, continue with the system generation procedure. The format of the restored disk is shown in Figure 17.

Real Cylinder	Number of Cylinders	Contents
0	1	VM/370 directory
1	1	191 minidisk for the IVPM1 user
2	1	191 minidisk for the IVPM2 user
3	1	191 minidisk for the REM2780 user
4-6	3	CP nucleus
7	1	Warm start data
8-9	2	I/O Error Recording area
10-33	24	Spooling and paging space
34	1	191 minidisk for the CPGEN user
35-144	110	190 minidisk (the CMS system disk) for the CMSSYS user Note: The last two cylinders of the 190 minidisk (143-144) contain the CMS nucleus.
154-202	58	194 minidisk of the CPGEN user - it contains the CP object modules (text decks)

Figure 17. Format of a 2314 Restored Disk

Figure 17 represents the allocation of areas on a 2314 starter system volume.

CPGEN, CMSSYS, IVPM1, IVPM2, and REM2780 are user identifications that own certain minidisks on the starter system volume.

CPGEN is the userid of the operator (that is, you use this userid to control the real system and to build a version of CP tailored to your installation.

CMSSYS is a directory entry on the starter system volume that owns CMSSYS 190 (the CMS system disk). CPGEN has a read-only link to CMSSYS 190 in order to use it to create the new system. Using this link, the CPGEN user can read from, but not write on, the 190 minidisk belonging to the CMSSYS user.

IVPM1 and IVPM2 are used with the Installation Verification Procedure to test the new system.

REM2780 is used to test the 2780 as a remote entry station.

SPECIAL CONSIDERATIONS FOR 800 BPI TAPES

If you ordered the starter system on 800 bpi tape, you have two tapes. If you have two tape drives available for the system generation procedure, use the following form of INPUT statement:

```
input 280 2400 281
```

In this case, when DDR finishes reading the first tape volume, it issues the message:

```
END OF VOLUME CYL xxx HD xx, MOUNT NEXT TAPE
```

and continues by reading the tape mounted on the alternate tape drive (281). If you have only one tape drive available, you enter an INPUT statement in the form:

```
input 280 2400
```

When the end-of-volume message appears, you must demount the first tape, mount the second tape on the same device and ready the device. Then, the DASD Dump Restore program continues by reading the second tape.

STEP 6. IPL THE STARTER SYSTEM

You now load (IPL) the starter system from the disk you restored it to. In our example, you press the Load pushbutton with the load unit address dials set to 130. At this point, only the device containing the system residence volume, 130, is known to the starter system.

STEP 7. DEFINE THE DEVICES NEEDED TO DO THE SYSTEM GENERATION

If your system console is at an address other than 009 or 01F, after you load the starter system you must press the Request key (or equivalent key) to enable the starter system to recognize the system console. If the console is not recognized, the VM/370 starter system enters a wait state.

At this point both the system residence volume and system console are recognized by the starter system and you can define the other devices you need. The starter system supports up to 16 channels, 8 control units, and 16 devices. The real control blocks for these devices are not built in the standard manner; the starter system builds them dynamically.

The starter system program (DMKSSP) then prompts you to answer the following questions until all the real control blocks necessary to operate a minimum machine configuration are created. The following example assumes: a 1403 printer at address 00E, a 2540 card reader/punch at addresses 00C (reader) and 00D (punch), tape drives at addresses 280 and 281, and a 2314 DASD at address 131. You must respond to the message

```
ENTER DEVICE ADDRESS WHERE SYSTEM RESIDENCE WILL BE BUILT (CUU):
```

with the same device address you specified on the SYSRES operand of the SYSRES macro in your system control file (DMKSYS).

If you are doing the system generation in a virtual machine, you must respond "no" to the message

```
CHANGE TOD CLOCK (YES|NO):
```

You must also respond to messages to set the date and time. If you are using a 3270 display device to generate VM/370 from the starter system for the first time, the 3270 locks when you try to respond to the message

```
SET DATE MM/DD/YY:
```

The System Available and Input Inhibited lights are on. You must press the Reset key and move the cursor to the second position of the input area before responding to the message. You follow this procedure only the first time the system is defined.

When asked what kind of start this is, you must respond "cold". The messages you receive at this time are:

```
VM/370 STARTER SYSTEM VERSION 2.0

ENTER PRINTER ADDRESS (CUU):00e
ENTER DEVICE TYPE (1403,1443,3211):1403
ENTER PUNCH ADDRESS (CUU):00d
ENTER DEVICE TYPE(2540P,3525):2540p
ENTER READER ADDRESS (CUU):00c
ENTER DEVICE TYPE (2501,2540R,3505):2540r
ENTER ADDRESS WHERE PID TAPE IS MOUNTED (CUU):280
ENTER DEVICE TYPE (2401,2415,2420,3420):2401
ENTER ADDRESS WHERE SCRATCH TAPE IS MOUNTED (CUU):281
ENTER DEVICE TYPE (2401,2415,2420,3420):2401
ENTER DEVICE ADDRESS WHERE SYSTEM RESIDENCE WILL BE BUILT (CUU):131
ENTER DEVICE TYPE (2319,2314,3330,2305):2314

***SYSTEM DEFINITION COMPLETED***

00E PRINTER
00D PUNCH
00C READER
280 PID TAPE
281 SCRATCH TAPE
131 NEW SYSTEM RESIDENCE
ARE THE ABOVE ENTRIES CORRECT (YES,NO):yes

VM/370 VERSION vv LEVEL 0 PLC 0000; mm/dd/yy hh:mm:ss
NOW 08:54:23 EDT FRIDAY mm/dd/yy
CHANGE TOD CLOCK (YES|NO):yes
SET DATE MM/DD/YY :mm/dd/yy
SET TIME HH:MM:SS :09:04:36
PRESS "TOD ENABLE SET" KEY AT DESIGNATED INSTANT

NOW 09:04:36 EDT FRIDAY mm/dd/yy
CHANGE TOD CLOCK (YES|NO):no
09:04:38 START ((COLD|WARM) (DRAIN))|(SHUTDOWN):cold
09:04:40
09:04:42 LOGON AT 09:04:37 EST FRIDAY mm/dd/yy
LINE 01F LOGON AS CPGEN      USERS= 001
09:04:42

DMKCPI952I nnnnK SYSTEM STORAGE
```

If you have not defined your system residence volume with a label of VMREL2, two messages are issued indicating that the volume labeled VMREL2 is not mounted. If you have labeled it as VMREL2, a message

indicating VMREL2 conflict is issued. You can ignore the following messages.

```
09:04:43 FILES:  NO RDR,  NO PRT,  NO PUN
09:04:44

DMKIOG556I FORMATTING I/O ERROR RECORDING MASK

09:04:46

DMKIOG557I FORMATTING MCH ERROR RECORDING AREA
```

The TOD clock referred to is the System/370 Time of Day clock. Enter the actual date and time in response to the "SET DATE" and "SET TIME" messages, and press the "TOD ENABLE SET" switch on the system control panel when the exact time specified agrees with the installation wall clock.

If, for some reason, you must reload the starter system, the following messages are displayed:

```
VM/370 STARTER SYSTEM VERSION 2.0

DO YOU WISH TO RE-DEFINE YOUR SYSTEM (YES|NO):
```

Respond by entering "no", unless the failure was the result of incorrectly specifying the device addresses, or unless the tapes and/or disks have been moved. If you reply "yes", you repeat Step 7.

STEP 8. SET THE TERMINAL MODE AND SPOOL THE CONSOLE

If the system console you are using is a display device, you should at this point, spool your console output so that you have a record of what you do. To spool the console input and output, issue the command:

```
spool console start
```

to save a copy of the system generation.

Because the default terminal environment for the primary system operator is CP, you should also issue the command:

```
terminal mode vm
```

The virtual machine terminal mode lets you remain in the CMS environment when you enter data on the display device.

STEP 9. DEFINE OR ATTACH THE SYSTEM RESIDENCE DEVICE

The CPGEN virtual machine assumes that the system residence volume is labeled VMREL2 and is at virtual address 350 if the device type is 2314 or at virtual address 351 if the device type is 3330. If you labeled your system residence volume VMREL2 in Step 2, your system residence device is already available; now you, as the operator of CPGEN, must define it. Use Procedure 1 to define your system residence device.

If you did not label your system residence device VMREL2, you must attach it to your virtual machine, CPGEN, now. Use Procedure 2 to attach your system residence device.

For example, if you used the values shown in Step 7, you designated the 131 drive, labeled VMREL2, as your system residence drive for the generation procedure. Now you must define your virtual device 350 so that it corresponds to the real disk 131.

Use one of the following procedures to define or attach your system residence volume:

- Procedure 1

  If you labeled your system residence volume VMREL2, but did not define its address as 350, you must do so now. Press the Request key and enter the command:

  ```
  define 350 as 131
  ```

  You, as the operator of the CPGEN virtual machine, gain ownership of that entire volume as virtual address 350. The CP DEFINE command maps this virtual address to the real address of your system residence volume as defined in your DMKSYS module.

  Remember that you restored your starter system tape to a 2314 device. If you are now creating a 3330 system residence, with label VMREL2, issue the command:

  ```
  define 351 as 131
  ```

  You cannot create a 3340 system residence directly from a 2314 starter system. However, a 3340 starter system is provided and a procedure for generating a 3340 system residence volume from a current 2314 or 3330 system residence volume and a PLC tape is described in Appendix H. VM/370 recommends that you use the 3340 starter system.

- Procedure 2

  If you did not label your system residence volume as VMREL2, you must attach your system residence volume to your virtual machine now. Press the Request key and enter the command:

  ```
  attach 131 to cpgen as 131
  ```

  Note: The first device address you specify is for the real device; the second device address is for the virtual device. The real 131 is the system residence volume you formatted in Step 2. The virtual 131 is the system residence device defined in DMKSYS (with the SYSRES macro) and also entered in response to the message:

  ```
  ENTER DEVICE ADDRESS WHERE SYSTEM RESIDENCE WILL BE BUILT (CUU):
  ```

STEP 10. MAKE OTHER DEVICES AVAILABLE

You must attach your tape drives and designate the printer to receive abnormal termination dumps if they occur. Press the Request key to enter each of the following commands:

```
attach 280 to cpgen as 181
attach 281 to cpgen as 182
set dump 00e
```

In the first command, the real tape drive (280) attached must be the same real address you entered in Step 7 in response to the message:

```
ENTER ADDRESS WHERE PID TAPE IS MOUNTED (CUU):
```

This real device must be attached to CPGEN as 181, because the VMFBLD EXEC procedure (which is used later) expects it to be 181.

In the second command, the real tape drive (281) attached must be the same real address you entered in Step 7 in response to the message:

```
ENTER ADDRESS WHERE SCRATCH TAPE IS MOUNTED (CUU):
```

This real device must be attached to CPGEN as 182, because the GENERATE EXEC procedure (which is used later) expects it to be 182.

If only one tape drive is available, enter

```
attach 280 to cpgen as 181
```

You can define your virtual 181 as 182 later in the system generation procedure.

The address of the real printer, defined in Step 7, is 00E. Any system dumps that occur are directed to that address.

If you wish, you can issue the CP command:

```
query virtual all
```

before and after performing Step 10, to see how your virtual machine's configuration changes.

STEP 11. LOAD CMS

Next, you must load CMS from virtual address 190 by issuing the CP command:

```
ipl 190
```

After CMS is loaded, a message is displayed on the system console indicating that CMS has successfully loaded:

```
CMS VERSION 2.0 mm/dd/yy hh.mm
```

Press the Return key and the CPGEN 191 minidisk is accessed as your A-disk.

STEP 12. LOAD THE PLC (PROGRAM LEVEL CHANGE) TAPE

Demount the PID (Program Information Department) starter system tape, and mount and ready the PLC tape at 280.

The PLC tape is prepared by the VM/370 Program Level Change (PLC) service. This is a system update service that can include new functions as well as cumulative system changes. The latest PLC tape contains all new updates, as well as all previous updates since the last VM/370 release base. IBM Field Engineering is responsible for initially ordering the PLC service. Thereafter, PID automatically ships the PLC tapes to the FE location, and FE is responsible for applying the updates to VM/370 systems.

The PLC tape supplied with the starter system should be installed as part of the system generation procedure. Issue the CMS command:

```
tape load
```

The system issues the following messages:

```
LOADING.....
VMFBLD   EXEC     A1
END-OF-FILE OR END-OF-TAPE
R;
```

The VMFBLD EXEC procedure is a service program that adds or replaces CP, CMS, or RSCS object modules. It loads the updates from the PLC tape for you. Issue the command:

```
access 191 c
```

because VMFBLD accesses other disks as its A-disk during processing. Next, issue the command:

```
vmfbld cp
```

to execute VMFBLD.

The VMFBLD EXEC prompts you with the message:

```
ENTER CP STAGING AREA DISK ADDRESS:
194
```

You respond with 194. The 194 minidisk belongs to the CPGEN virtual machine and is the area on the restored starter system volume that contains all the object modules necessary to generate a CP nucleus. The contents of the minidisk are updated with the latest versions of all object modules, as well as the latest version of the DMKMAC macro library. The following message is displayed:

```
ARE YOU A NEW USER? -- RESPOND (YES|NO)
```

You must respond "yes" to this question. The following messages are then displayed:

```
'190' REPLACES 'A(194)'
190 ALSO = S-DISK
hh:mm:ss REWIND COMPLETE
CMS VERSION v.0 - mm/dd/yy hh.mm
```

The VMFBLD EXEC procedure links to the CMS system disk and accesses it as the A-disk. Next, VMFBLD reads the updated service programs from the PLC tape and writes them to the CMS S-disk. Then VMFBLD loads the new CMS system onto the system disk.

After CMS is loaded, a message is displayed to indicate that the CMS system was successfully loaded:

```
CMS VERSION v.0 - mm/dd/yy hh.mm
```

You must enter enter a null line to continue.

STEP 13. PUNCH THE SERVICE PROGRAMS

Use the GENERATE EXEC procedure to punch the service programs. Issue the command:

```
generate srvcpgm
```

These service programs are needed for standalone use; you should externally identify the decks and keep them intact.

The programs punched are:

- DMKFMT (a three-card loader precedes the DMKFMT text deck)
- DMKDIR (a three-card loader precedes the DMKDIR text deck)
- DMKDDR (a three-card loader precedes the DMKDDR text deck)
- IBCDASDI (the IBCDASDI text deck is loadable)

The GENERATE EXEC issues the following message:

```
THE FOLLOWING STANDALONE SERVICE PROGRAMS ARE BEING PUNCHED
      ** FORMAT - DIRECT - DUMP/RESTORE - IBCDASDI **

PUNCHING ' IPL FMT ' ******

PUNCHING ' IPL DIR ' ******

PUNCHING ' IPL DDR ' ******

PUNCHING ' IPL IBCDASDI ' ******
```

Each program deck is preceded by a CP userid card and several separator cards, all of which may be discarded. The format of these cards is described in the VM/370: Operator's Guide.

For more information about the GENERATE EXEC procedure, see "Part 6: Updating VM/370."

STEP 14. PRINT AND PUNCH THE STARTER SYSTEM SUPPLIED DIRECTORY, DMKSNT, DMKSYS, AND DMKFCB

After the service programs are punched, the GENERATE EXEC asks you whether you want a copy of the Release 2 directory printed. You should respond "yes."

```
PRINT COPY OF RELEASE2 DIRECT? -- RESPOND (YES|NO): yes
```

This prints a copy of the Release 2 directory, as well as copies of the DMKSYS ASSEMBLE (the CP system control file), DMKFCB ASSEMBLE (the forms control buffer file), and DMKSNT ASSEMBLE (the system name table) provided with the starter system.

The GENERATE EXEC procedure issues the following message:

```
A SAMPLE DIRECTORY IS BEING PRINTED TO AID YOU.
IT SHOWS WHERE THE VIRTUAL DISKS ARE LOCATED ON 'CPV2L0'
YOU MAY USE THESE MINIDISKS FOR OTHER VIRTUAL MACHINES,
IN PARTICULAR THE CMS SYSTEM DISK ( MAINT 190 ) AND
THE CP STAGING AREA DISK ( MAINT 194 )
INCLUDED IN THIS DIRECTORY IS THE USERID: MAINT
WHICH WILL BE USED FOR FUTURE SUPPORT OF THE SYSTEM.
THIS USERID SHOULD BE INCLUDED IN THE DIRECTORY
YOU BUILD FOR YOUR FLOOR USE.
** CAUTION ** IF YOU DESTROY USER MAINT'S AREAS, IT WILL BE
              NECESSARY TO RE-BUILD THE ENTIRE SYSTEM.
```

```
A SAMPLE OF DMKSYS, DMKFCB, AND DMKSNT ASSEMBLE ARE ALSO BEING
PRINTED TO AID YOU. THIS SAMPLE DMKSNT IS BASED ON THE
INFORMATION INCLUDED IN THE SAMPLE DMKSYS AS WELL AS THE
EXAMPLE ALLOCATIONS FOR VMREL2 PROVIDED IN THE SYSGEN GUIDE.
A COPY OF THIS DMKSNT MODULE HAS BEEN INCLUDED IN THE CP NUCLEUS,
SUCH THAT IF ONE USES THE INCLUDED DMKSYS AND THE
SAMPLE ALLOCATION PROVIDED IN THE SYSTEM GENERATION GUIDE,
HE WILL BE ABLE TO SAVE HIS CMS SYSTEM UPON COMPLETION
OF THE SYSTEM GENERATION PROCEDURE. A COPY OF DMKFCB HAS BEEN
INCLUDED IN THE NUCLEUS AND NEED NOT BE RE-ASSEMBLED FOR
SYSTEM GENERATION. IT HAS BEEN INCLUDED FOR THE USER WHO WOULD LIKE
TO MODIFY OR ADD TO THE EXISTING BUFFER LOAD.

NOTE: IF THE USER WISHES TO MODIFY THE SAMPLE DMKSNT AND/OR DMKFCB
HE MAY INCLUDE THE UPDATED SOURCE WITH THE SOURCE INCLUDED UNDER
THE OPTION 'GENERATE VM370', OF THE SYSTEM GENERATION PROCEDURE.
IF PRESENT, IT WILL AUTOMATICALLY BE ASSEMBLED AND INCLUDED IN THE
NEW CP NUCLEUS.
```

Again, the GENERATE EXEC procedure prompts you:

```
DO YOU WISH TO HAVE A COPY OF DMKSNT, DMKSYS, DMKFCB, AND
RELEASE2 DIRECT PUNCHED TO CARDS? -- RESPOND (YES|NO):
```

You should enter "yes" to have these decks punched. "Part 2: Defining Your VM/370 System" contains a listing of the Release 2 directory and the DMKSNT, DMKSYS, and DMKFCB modules supplied with the starter system. The following messages are issued to indicate the files that are being punched:

```
PUNCHING ' DMKSNT ASSEMBLE ' ******

PUNCHING ' DMKSYS ASSEMBLE ' ******

PUNCHING ' DMKFCB ASSEMBLE ' ******

PUNCHING ' RELEASE2 DIRECT ' ******
R;
```

STEP 15. BUILD THE VM/370 DIRECTORY AND ASSEMBLE THE FILES DEFINING THE REAL I/O AND SYSTEM DEVICES

Next, you should place the files describing your installation's version of the VM/370 directory, real I/O configuration, and system devices in the reader and invoke the GENERATE EXEC procedure to build the directory and assemble the files describing the configuration. Place the following cards in the card reader, in the sequence shown:

```
ID CPGEN
:READ  filename DIRECT
(Directory program control statements)
:READ  DMKRIO ASSEMBLE
(Real I/C configuration macros)
:READ  DMKSYS ASSEMBLE
(CP system control file macros)
```

The Directory program control statements, real I/O configuration macros, and CP system control file macros are those you created according to the instructions in "Part 2: Defining Your VM/370 System." You can code your own system control macros or add to the file supplied with the starter system.

Also, if you want, you can change the printer buffer load (DMKFCB) or the named system (DMKSNT) modules to conform to local requirements. If you want to change the printer buffer load or the system name table, you should add your changes to the files already punched (in Step 14) and place the following files in the card reader behind the System Control file macros:

```
:READ DMKFCB ASSEMBLE
(forms control macros)
:READ DMKSNT ASSEMBLE
(system name table macros)
```

Instructions for creating new forms control macros are in the VM/370: System Programmer's Guide. The system name table macros are described in "Part 2: Defining Your VM/370 System" of this manual.

FORMAT OF THE READ CONTROL STATEMENT

The READ control statements must be punched according to the following format:

Column	Number of Characters	Contents	Meaning
1	1	colon':'	Identifies card as a control card
2-5	4	READ	Identifies card as a READ control card
6-7	2	blank	
8-15	8	fname	Filename of the file punched
16	1	blank	
17-24	8	ftype	Filetype of the file punched
25-80	56	blank	

SPECIAL PROCEDURE IF YOU ARE USING ONLY ONE TAPE DRIVE

If you are using only one tape drive, at this time you must issue the command:

```
define 181 as 182
```

Now mount the scratch tape in place of the PLC tape and ready the device.

INVOKE THE GENERATE EXEC PROCEDURE

When all the files are placed in the reader, invoke the GENERATE EXEC procedure by issuing the following command:

```
generate vm370
```

This procedure invokes the Directory service program to build the disk-resident VM/370 directory, then assembles the DMKRIO and DMKSYS files that you placed in the real card reader. After the Directory service program execution completes, the DMKRIO and DMKSYS files are assembled in preparation for building the new CP system nucleus. GENERATE then checks for DMKFCB and DMKSNT source files. When new versions of the DMKFCB and DMKSNT modules are provided, GENERATE assembles the new modules and replaces the corresponding modules supplied with the starter system with the new modules. If any errors occur while the VM/370 directory is being built, the Directory program issues error messages and the GENERATE EXEC procedure issues the following message:

```
CORRECT THE DIRECTORY CARDS AND RELOAD THE CARD READER
  RESPOND WITH: GENERATE DIRECT
```

Correct the errors in the Directory program control statements, and reload the card reader with only the ID card, the :READ statement, and the Directory program control statements. Then respond with:

```
generate direct
```

If errors are detected while the DMKRIO, DMKSYS, DMKFCB, or DMKSNT files are assembling, GENERATE issues a similar message:

```
CORRECT THE filename ASSEMBLE FILE AND RELOAD THE CARD READER
  RESPOND WITH: GENERATE filename
```

You must correct the errors in the indicated file, and reload the card reader with only the ID card, the :READ statement, and the appropriate file. Then, issue the GENERATE command with the appropriate option. For example:

```
generate dmksys
```

STEP 16. SPECIAL CONSIDERATIONS FOR VIRTUAL=REAL MACHINES

Once the directory is built and the files are assembled, the GENERATE EXEC procedure asks if you want the virtual=real option included in your CP nucleus:

```
VIRTUAL=REAL OPTION REQUIRED (YES,NO):
```

If you respond "yes" to this question, you are prompted to enter the amount of storage you wish to reserve in the CP nucleus for a virtual=real machine. ("Part 1. Planning for System Generation" contains formulas to help you determine how much storage you need to reserve.) This size must be a multiple of 4K and must be entered in the format nnnnK. The minimum size you can specify is 32K and the maximum is 4M. For example:

```
STORAGE SIZE OF VIRT=REAL (MINIMUM IS 32K):
100k
0100K STORAGE SIZE FOR VIRTUAL=REAL
IS THE ABOVE ENTRY CORRECT (YES,NO):
yes
```

STEP 17. LOAD THE CP NUCLEUS

Once you respond to the virtual=real generation questions, the GENERATE EXEC procedure writes the CP nucleus on tape. To do this, GENERATE first issues the the VMFLOAD command to punch the loader and CP object modules to a virtual punch spool file. Then it transfers this file to the virtual card reader file and writes the file on tape. GENERATE issues the following messages during the processing:

```
hh:mm:ss NO FILES PURGED
SYSTEM LOAD DECK COMPLETE
hh:mm:ss PUN FILE nnnn TO CPGEN COPY 01 NOHOLD
IPLABLE NUCLEUS NOW ON TAPE ****
hh:mm:ss PRT cuu OUTPUT OF CPGEN FILE=nnnn RECDS=nnnnnn
  COPY = 01 A PRT
```

Note: If any errors are detected while the tape is being written, you must recreate the CP nucleus. To do this, issue the command:

```
generate cp nucleus
```

The procedure restarts at Step 16 where you are asked if you want the virtual=real option.

LOADING A CP NUCLEUS WITHOUT A VIRTUAL=REAL AREA

If you responded "no" when asked if you wanted the virtual=real area, the GENERATE EXEC procedure loads the nucleus for you. You receive the following message:

```
WHEN 'NUCLEUS LOADED ON XXXXXX' IS TYPED, ISSUE 'CLOSE PRT',
  TO GET THE CPLOAD MAP.  WHEN PRINTING IS COMPLETE,
  SHUTDOWN THE SYSTEM AND IPL THE NEW SYSRES VOLUME.
```

When the nucleus is written on the system residence volume, the message

```
NUCLEUS LOADED ON volid
```

is issued, where volid is the volume serial number of your system residence volume. The volid is the serial number you specified on the SYSRES macro when you prepared the CP system control file (DMKSYS). If you followed the example in this manual, the serial number of your system residence volume is VMREL2.

To print the load map you must close the printer; issue the command:

```
close prt
```

When printing is complete, shutdown the system in preparation for loading your newly generated CP nucleus:

```
shutdown
```

LOADING A CP NUCLEUS THAT HAS A VIRTUAL=REAL AREA

If you responded "yes" when asked if you wanted the virtual=real area, the GENERATE EXEC procedure issues the following message:

TO LOAD THE CP NUCLEUS JUST CREATED, SHUTDOWN THE SYSTEM AND THEN IPL THE TAPE. THE CPLOAD MAP WILL AUTOMATICALLY BE PRINTED AT THE PRINTER WHOSE ADDRESS IS '00E'. IF THERE IS NO PRINTER AT THIS ADDRESS THE LOAD MAP WILL BE PRINTED AT THE FIRST PRINTER CAUSING AN INTERRUPT, (IE. NOT-READY TO READY SEQUENCE). ONCE THE NUCLEUS HAS BEEN LOADED, YOU MAY IPL YOU NEW CP SYSTEM RESIDENCE VOLUME.

NOTE: THERE MUST BE ENOUGH STORAGE ON THE SYSTEM (VIRTUAL OR REAL), TO CONTAIN THE VIRT=REAL AREA AND THE CP NUCLEUS.

You must be sure that there is enough storage to load the CP nucleus with a virtual=real area before you load it. The "Specifying a Virtual=Real Machine" section of Part 1 tells you how to determine the amount of real or virtual storage you need. You can load a CP nucleus that has a virtual=real area in either a real or virtual machine.

You IPL the tape containing the CP nucleus next. The CP nucleus is on the tape at virtual address 182 for the CPGEN virtual machine.

STEP 18. IPL THE NEWLY GENERATED VM/370

IPL the newly created system residence volume by setting the load unit address dials to the address of the system residence volume and pressing the Load pushbutton. The userid OPERATOR (or whatever userid you specified as the system operator on the SYSOPER macro) is logged on when you IPL. The following messages are issued. Respond as shown.

```
VM/370 VERSION vv LEVEL 00 PLC nnnn; mm/dd/yy hh:mm:ss

NOW hh:mm:ss EDT day mm/dd/yy
CHANGE TOD CLOCK (YES|NO):no
hh:mm:ss START ((COLD|WARM) (DRAIN))|(SHUTDOWN): cold
hh:mm:ss LOGON AT hh:mm:ss EDT day mm/dd/yy
hh:mm:ss LINE 01F LOGON  AS OPERATOR USERS = 001
hh:mm:ss

DMKCPI952I nnnnK SYSTEM STORAGE

hh:mm:ss FILES:  NO RDR,  NO PRT,  NO PUN
hh:mm:ss

DMKIOG556I FORMATTING I/O ERROR RECORDING AREA

hh:mm:ss

DMKIOG557I FORMATTING MCH ERROR RECORDING AREA
```

At the time you IPL your newly generated VM/370, your system residence volume is formatted according to your specification on the SYSRES macro. Also, if you used the VM/370 directory supplied with the 2314 starter system, your starter system volume (CPV2L0) is set up as shown in Figure 18. Be careful not to modify the 190 and 194 minidisks for the MAINT user. These minidisks and the information they contain are required for applying PLC (Program Level Change) updates to VM/370.

Real Cylinder	Number of Cylinders	Contents
0	1	Unused
1	1	191 minidisk for the IVPM1 user
2	1	191 minidisk for the IVPM2 user
3	1	191 minidisk for the REM2780 user
4-13	10	191 minidisk for the OPERATOR user
14-18	5	191 minidisk for the CE user
19-28	10	191 minidisk for the MAINT user
29-33	5	191 minidisk for the ECMODE user
34	1	199 minidisk for the MAINT user
35-144	110	190 minidisk for the MAINT user Note that the last two cylinders of this minidisk contain the CMS nucleus.
145-202	58	194 minidisk for the MAINT user - it contains the CP object modules (text decks)

Figure 18. Allocation of the VM/370 Starter System Volume (CPV2L0) when the 2314 Starter System Directory Is Used

STEP 19. BACK UP THE NEWLY GENERATED VM/370

At this time, you should be sure to back up your new system residence volume. If your real machine has at least 448K bytes of real storage, the tape created in Step 17 is sufficient backup. However, that tape is not sufficent backup for real systems with less than 448K of real storage because that tape cannot be loaded on such systems.

VM/370 systems that run on a real machine with less than 448K bytes of real storage should use the DASD Dump Restore (DDR) service program to create a backup tape similar to the one created in Step 17. The DASD Dump Restore program is described in Appendix F. If your system residence volume is at address 131 and you labeled it VMREL2, you could use the following DDR control statements to back it up:

```
input 131 2314 vmrel2
output 180 3420
dump cpvol
```

The tape produced contains the CP nucleus; it does not contain the VM/370 directory. However, you already created a standalone version of the Directory program in Step 13 and have the Directory program control statements, which you used in Step 15 to recreate the VM/370 directory.

If you do not wish to use the DDR program to backup your system, you can load the tape produced in Step 17 in a virtual machine. If you load

| the tape in a virtual machine that virtual machine must have (1) 512K of
| storage and (2) write-access to the system residence volume, at the
| address defined for system residence in the SYSRES macro of the CP
| system control (DMKSYS) file.

| When you use the DDR program to backup your system, you do not get a
| load map when you restore the tape. You do get a load map if you load
| the tape produced in Step 17.

STEP 20. MOVE THE CMS SYSTEM DISK, IF YOU WISH

The CP nucleus just created is tailored to your installation's exact I/O configuration with all the appropriate options selected. The CMS nucleus requires no special tailoring for installation. The CMS system residence disk was created when the VM/370 starter system was loaded; it contains a copy of the CMS nucleus and disk-resident command modules. At this point, the CMS system resides on the virtual 190 disk for the user, MAINT. The operator's VM/370 directory has an entry that links to MAINT's 190 disk, therefore you can invoke CMS by issuing the command:

```
ipl 190
```

| and thus have access to most of the CMS functions. Although most of the
| CMS functions are available, CMS has not yet been updated with the
| latest PLC changes. The assembler that is available now is the one
| supplied with the starter system.

If you wish, you can move the CMS system disk to a disk other than the one designated by the starter system. To move the CMS system disk to another minidisk, use the COPY function of the DDR service program. The new minidisk should be the same size as the CMS system disk, but it does not need to be formatted. Also, if you move the CMS system, you must make the appropriate changes to the userid MAINT in your VM/370 directory (that is, change the volume label on the MDISK entry for 190).

If you add IBM Program Products or your own disk-resident modules to the CMS system disk, you may have to create a new system disk to contain the additional modules. The "Creating a CMS System Disk" section of "Part 6. Updating VM/370" tells you how to do this.

STEP 21. FORMAT THE OPERATOR'S VIRTUAL 191 DISK

Before any new minidisk area can be used for CMS files, it must be initialized with the CMS FORMAT command, which formats the area into 800-byte blocks. Take care not to format areas which contain data restored from the starter system (such as the 190 and 194 minidisks belonging to the user MAINT.) The CMS FORMAT command is described in the *VM/370: Command Language Guide for General Users*.

Note: After you complete this step, a portion of the starter system is overlaid by the operator's 191 minidisk. If for any reason you wish to IPL the starter system again, you must restart at step 1.

At this time you are logged on as the operator. Use the following procedure to format your virtual disk 191. First, load CMS (IPL 190). CMS responds with:

```
CMS VERSION 2.0 - mm/dd/yy hh:mm
```

Next, enter the following command:

```
access (nodisk
```

The NODISK option prevents CMS from automatically accessing your virtual disk 191. (Accessing 191 at this time would causes an error message to be issued because 191 is not yet initialized, and therefore cannot be used.) After the Ready message is displayed, issue the command:

```
format 191 a
```

The CMS FORMAT command prompts you with the following message:

```
FORMAT WILL ERASE ALL FILES ON DISK 'A(191)'. DO YOU WISH TO
  CONTINUE? (YES|NO):
```

If you respond "yes", CMS prompts you with:

```
ENTER DISK LABEL:
opr191
```

Enter the one- to six-character alphameric label of the virtual disk. You can use whatever label you wish for this virtual disk. In this example, the label is OPR191. CMS then issues:

```
FORMATTING DISK 'A'.
'10' CYLINDERS FORMATTED ON 'A(191)'.
```

and a Ready message.

STEP 22. FORMAT THE MAINT USER'S VIRTUAL 191 DISK

Next, you should initialize the 191 disk belonging to MAINT, in the same way you initialized the operator's virtual 191. Log yourself off the system and log on again as userid MAINT, using the password CPCMS. Now IPL 190. When the CMS Ready message is displayed, issue

```
access (nodisk
```

and continue to format MAINT's 191, using the same procedure you used to format OPR191.

STEP 23. UPDATE CMS

Now that the MAINT 191 minidisk is formatted, you (as the MAINT user) can update CMS with the changes supplied on the PLC tape. Mount the PLC tape, if it is not already mounted. You must attach the real tape drive to your CMS virtual machine (MAINT); for example:

```
attach 280 to maint as 181
```

and, with the 191 minidisk that belongs to MAINT accessed as your A-disk, rewind and load the tape:

```
tape rew
tape load
```

CMS responds with the following message:

```
LOADING.....
VMFBLD   EXEC   A
END-OF-FILE OR END-OF-TAPE
```

BUILDING A NEW CMS

Now execute the VMFBLD EXEC procedure by issuing the commands:

```
access 191 c
vmfbld cms
```

The VMFBLD EXEC procedure prompts you with:

```
ENTER CMS SYSTEM DISK ADDRESS:
```

You respond with "190". Next, VMFBLD issues the message:

```
190 ALSO = S-DISK.
```

| You are asked if you want the service programs punched:

```
| PUNCH STANDALONE SERVICE PROGRAMS--RESPOND (YES|NO):
```

The following messages are displayed; respond as shown:

```
hh:mm:ss  NO  FILES PURGED
hh:mm:ss  NO  FILES PURGED
SYSTEM LOAD DECK COMPLETE
hh:mm:ss PUN FILE nnnn  TO  CPGEN    COPY 01 NOHOLD
hh:mm:ss REWIND COMPLETE
         WHEN THE NEW CMS SYSTEM IS BUILT ISSUE THE FOLLOWING
         COMMAND
              CLOSE PRT . . . (PRINTS THE LOAD MAP)

DMSINI606R SYSTEM DISK ADDRESS = 190
DMSINI615R Y-DISK ADDRESS = 19e
DMSINI607R REWRITE THE NUCLEUS ? yes
DMSINI608R IPL DEVICE ADDRESS = 190
DMSINI609R NUCLEUS CYL ADDRESS = 108
DMSINI610R ALSO IPL CYLINDER 0 ? yes
DMSINI611R VERSION IDENTIFICATION =
DMSINI612R INSTALLATION HEADING =
CMS VERSION 2.0 -mm/dd/yy hh:mm
```

In the example, the 190 you entered is the 190 minidisk that belongs to the user MAINT; it is equivalent to the 190 minidisk that belongs to CPGEN on the starter system. If you used the VM/370 directory supplied with the starter system, you allocated 110 cylinders (virtual cylinders 0-109) to the 190 minidisk belcnging to the user MAINT. The last two cylinders of this minidisk contain the CMS nucleus, therefore you specify cylinder 108 as the nucleus cylinder address when responding to message DMSINI609R.

When the VMFBLD EXEC procedure completes its execution, the CMS system residence volume is updated with the most current object modules (text decks) and load modules, and the new CMS nucleus is written on the CMS system residence volume.

Also, if there are any updates for the EREP program or System Assembler on the PLC tape, the VMFBLD EXEC procedure updates those programs and creates the corresponding new auxiliary directory.

STEP 24. SAVE CMS

If you used the sample DMKSNT and DMKSYS supplied with the starter system, and used the sample allocations shown in Step 2, you can now save your CMS system.

To save the CMS system, you must load it and then save it as soon after loading as possible. Issue the command:

```
ipl 190
```

When the terminal unlocks, do not press carriage return, but immediately issue the command:

```
savesys cms
```

Then press carriage return. If CMS is successfully saved, the message

```
hh:mm:ss SYSTEM SAVED
CMS VERSION 2.0 - mm/dd/yy hh:mm
```

is displayed. Your CMS system is now saved; you can issue IPL CMS instead of IPL 190, when you wish to run CMS.

If you named your CMS system something other than CMS, such as CMS1, the entry you made in the system name table would be for CMS1, you would have to save CMS1 (SAVESYS CMS1) and then you could IPL CMS1. For more information about how to save CMS, see the VM/370: System Programmer's Guide or the "Saved Systems" section of this manual.

At this point, use the Installation Verification Procedure (IVP) to test the new system. Log off as the userid MAINT and log on again as the userid OPERATOR, using the password OPERATOR. The IVP is described later in Part 3.

Generating CP and CMS Using the 3330 Starter System

Except where otherwise noted, you can, if you wish, substitute other values in place of the device addresses, volume labels, and allocations shown. Note that if you use the sample DMKSNT and DMKSYS provided with the starter system, and use the sample allocations shown in Steps 2 and 3, you can save your CMS system at the end of the procedure.

It is strongly recommended that you use the sample allocations given in Step 2 and the label VMREL2 for the new system residence volume, to ensure that you have sufficient TEMP space to complete the system generation. (The TEMP space provided on the starter system volume may not be sufficient for large systems.)

| Note: The 3330 starter system cannot be generated on a real System/370 | with a console address greater than 5FF hexadecimal.

The examples of messages and responses assume that you are performing the system generation at a typewriter terminal, such as a 3210, 2741, or 3767 (operating as a 2741). If you are using a display device, such as the 3277, when you type the response to a prompting message, the response appears in the user input area. When you enter that response, it is redisplayed in the output area on the line below the prompting message. Also, if the standalone service programs (such as the DASD Dump Restore program or Format/Allocate program) send output to a terminal display screen, the output is wrapped around immediately, when the screen becomes full, to continue displaying.

While you are generating the system, you may see some extraneous messages as the starter system is processing. These are not shown in the examples below. Only those messages that you should take note of or respond to are shown.

STEP 1. LOAD THE FORMAT PROGRAM FROM THE STARTER SYSTEM TAPE

| Mount the CP starter system tape and IPL the tape. The CP Format/Allocate service program is the first file on the tape; it is now loaded. Do not rewind the tape because the next file is needed later in the system generation procedure (Step 4).

STEP 2. FORMAT, LABEL, AND ALLOCATE THE SYSTEM RESIDENCE VOLUME

| Use the CP Format/Allocate program to format, label, and allocate space on the new system residence volume. First, identify the system console by pressing the Request key (or equivalent); if the console address is either 009 or 01F, you do not have to press the Request key. Then, to execute the Format/Allocate service program, respond to the prompting messages.

In the following example, the responses (format, 350, 3330, 000, 403, and vmrel2) format the 3330 disk at address 350 and label it VMREL2. Whatever label you specify must match what you specified on the SYSVOL operand of the SYSRES macro when you defined the system. In any case, do not use CPV2L0 because that is the label of the starter system disk. The console output displays:

```
VM/370 FORMAT/ALLOCATE PROGRAM VERSION 2.0

ENTER FORMAT OR ALLOCATE:format
FORMAT FUNCTION SELECTED
ENTER DEVICE ADDRESS (CCU):350
ENTER DEVICE TYPE:3330
ENTER START CYLINDER (XXX) OR "LABEL":000
ENTER END CYLINDER (XXX):403
ENTER DEVICE LABEL:vmrel2
FORMAT STARTED
FORMAT DONE
000 PAGE RECORDS FLAGGED
```

When the format operation completes, the prompting message

```
ENTER FORMAT OR ALLOCATE:
```

is displayed. Now that the system residence volume is formatted and labeled, you must allocate the disk space. Again, you must respond to the prompting messages. In the following example, the space on the 3330 disk at address 350, with the label VMREL2, is allocated (7 cylinders of permanent space, 4 cylinders of directory, 379 cylinders of temporary space, and 14 cylinders of TDSK space). You can use the formulas given in the "Creating Your VM/370 Directory" section of Part 2 to ensure that four cylinders give enough space for your VM/370 directory. If you do not allocate your DASD space as shown in this example, you are responsible for ensuring that you have enough temporary space to perform the assemblies associated with VM/370 system generation.

```
ENTER FORMAT OR ALLOCATE:allocate
ALLOCATE FUNCTION SELECTED
ENTER DEVICE ADDRESS(CCU):350
ENTER DEVICE TYPE:3330
ENTER DEVICE LABEL:vmrel2
ENTER ALLOCATION DATA FOR VOLUME VMREL2
TYPE CYL CYL
.... ... ...
perm 000 006
drct 007 010
temp 011 389
tdsk 390 403
end
ALLOCATION RESULTS
PERM 000 006
DRCT 007 010
TEMP 011 389
TDSK 390 403
DEVICE 350 VOLUME VMREL2 ALLOCATION ENDED
ENTER FORMAT OR ALLOCATE:
```

STEP 3. LABEL THE STARTER SYSTEM VOLUME

Use the Format/Allocate program to label the scratch volume that is to contain the CP starter system. This label must be CPV2L0. You can format and label this volume, or just label it. Formatting is unnecessary because you are going to restore the starter system to this volume. In the following example, the responses (format, 351, 3330, label, and cpv2l0) to the prompting messages put the label CPV2L0 on the 3330 disk at address 351.

```
ENTER FORMAT OR ALLOCATE: format
FORMAT FUNCTION SELECTED
ENTER DEVICE ADDRESS (CCU):351
ENTER DEVICE TYPE:3330
ENTER START CYLINDER (XXX) OR "LABEL":label
ENTER DEVICE LABEL: cpv2l0
LABEL IS NOW CPV2L0
```

When the Format/Allocate program is complete, it responds:

```
ENTER FORMAT OR ALLOCATE:
```

You do not have to respond to this message.

Now the volume is available and ready for the data that is to be placed on it by the DASD Dump Restore service program (module DMKDDR). The DASD Dump Restore program is the second file on the starter system tape.

STEP 4. LOAD THE DASD DUMP RESTORE PROGRAM FROM THE STARTER SYSTEM TAPE

IPL the starter system tape a second time to load the DASD Dump Restore (DDR) program. It is the second file on the starter system tape. Again, do not rewind the tape because the next file is needed in Step 5.

STEP 5. RESTORE THE STARTER SYSTEM TO DISK

Respond to the DDR prompting messages to restore the starter system.

In the following example, the starter system is restored from the 2400 series tape drive at address 280 to the 3330 disk at address 351 (and with label CPV2L0). The console output is:

```
VM/370 DASD DUMP/RESTORE PROGRAM VERSION 2.0
ENTER CARD READER ADDRESS OR CONTROL STATEMENTS
ENTER: sysprint cuu (cuu=real printer address)
ENTER: input 280 2400 (See the "Special Considerations for 800 BPI Tapes"
       section, if you ordered your starter system on 800 bpi tape)
ENTER: output 351 3330 cpv2l0
ENTER: restore all
RESTORING CPV2L0
END OF RESTORE
ENTER: (null line -- END key on 3215 or Enter key on 3277)
END OF JOB
```

The DDR program restores the third file on the starter system tape to the disk labeled CPV2L0. The restored disk contains:

- A VM/370 directory (RELEASE2 DIRECT), as well as sample source for DMKSYS, DMKSNT, and DMKFCB
- A VM/370 starter system nucleus
- A complete CMS system residence volume
- A complete CP system comprised of macro libraries and text files

When the disk is restored, continue with the system generation procedure. The format of the restored disk is shown in Figure 19.

Real Cylinder	Number of Cylinders	Contents
0	1	VM/370 directory
1	1	191 minidisk for the IVPM1 user
2	1	191 minidisk for the IVPM2 user
3	1	191 minidisk for the REM2780 user
4–5	2	CP nucleus
6	1	Warm start data
7–8	2	I/O Error Recording area
9–28	20	Spooling and paging space
29	1	191 minidisk for the CPGEN user
30–105	76	190 minidisk (the CMS system disk) for the CMSSYS user Note: The last cylinder of the 190 minidisk (105) contains the CMS nucleus.
106–149	44	194 minidisk of the CPGEN user – it contains the CP object modules (text decks)
150–403	254	Not used.

Figure 19. Format of a 3330 Restored Disk

Figure 19 represents the allocation of areas on a 3330 starter system volume.

CPGEN, CMSSYS, IVPM1, IVPM2, and REM2780 are user identifications that own certain minidisks on the starter system volume.

CPGEN is the userid of the operator (that is, you use this userid to control the real system and to build a version of CP tailored to your installation.

CMSSYS is a directory entry on the starter system volume that owns CMSSYS 190 (the CMS system disk). CPGEN has a read-only link to CMSSYS 190 in order to use it to create the new system. Using this link, the CPGEN user can read from, but not write on, the 190 minidisk belonging to the CMSSYS user.

IVPM1 and IVPM2 are used with the Installation Verification Procedure to test the new system.

REM2780 is used to test the 2780 as a remote entry station.

SPECIAL CONSIDERATIONS FOR 800 BPI TAPES

If you ordered the starter system on 800 bpi tape, you have two tapes. If you have two tape drives available for the system generation procedure, use the following form of the INPUT statement:

```
input 280 2400 281
```

In this case, when DDR finishes reading the first tape volume, it issues the message:

```
END OF VOLUME CYL xxx HD xx, MOUNT NEXT TAPE
```

and continues by reading the tape mounted on the alternate tape drive (281). If you have only one tape drive available, you enter an INPUT statement in the form:

```
input 280 2400
```

When the end-of-volume message appears, you must demount the first tape, mount the second tape on the same device and ready the device. Then, the DASD Dump Restore program continues by reading the second tape.

STEP 6. IPL THE STARTER SYSTEM

You now load (IPL) the starter system from the disk you restored it to. In our example, you press the Load pushbutton with the load unit address dials set to 351. At this point, only the device containing the system residence volume, 351, is known to the starter system.

STEP 7. DEFINE THE DEVICES NEEDED TO DO THE SYSTEM GENERATION

If your system console is at an address other than 009 or 01F, after you load the starter system you must press the Request key (or equivalent key) to enable the starter system to recognize the system console. If the console is not recognized, the VM/370 starter system enters a wait state.

At this point both the system residence volume and system console are recognized by the starter system and you can define the other devices you need. The starter system supports up to 16 channels, 8 control units, and 16 devices. The real control blocks for these devices are not built in the standard manner; the starter system builds them dynamically.

The starter system program (DMKSSP) then prompts you to answer the following questions until all the real control blocks necessary to operate a minimum machine configuration are created. The following example assumes: a 1403 printer at address 00E, a 2540 card reader/punch at addresses 00C (reader) and 00D (punch), tape drives at addresses 280 and 281, and a 3330 DASD at address 350. You must respond to the message

```
ENTER DEVICE ADDRESS WHERE SYSTEM RESPONSE WILL BE BUILT (CUU):
```

with the same device address you specified on the SYSRES operand of the SYSRES macro in your system control file (DMKSYS).

If you are doing the system generation in a virtual machine, you must respond "no" to the message:

```
CHANGE TOD CLOCK (YES|NO):
```

You must also respond to messages to set the date and time. If you are using a 3270 display device to generate VM/370 from the starter system for the first time, the 3270 locks when you try to respond to the message

```
SET DATE MM/DD/YY:
```

The System Available and Input Inhibited lights are on. You must press the Reset key and move the cursor to the second position of the input area before responding to the message. You follow this procedure only the first time the system is defined.

When asked what kind of start this is, you must respond "cold". The messages you receive at this time are:

```
VM/370 STARTER SYSTEM VERSION 2.0

ENTER PRINTER ADDRESS (CUU):00e
ENTER DEVICE TYPE (1403,1443,3211):1403
ENTER PUNCH ADDRESS (CUU):00d
ENTER DEVICE TYPE(2540P,3525):2540p
ENTER READER ADDRESS (CUU):00c
ENTER DEVICE TYPE (2501,2540R,3505):2540r
ENTER ADDRESS WHERE PID TAPE IS MOUNTED (CUU):280
ENTER DEVICE TYPE (2401,2415,2420,3420):2401
ENTER ADDRESS WHERE SCRATCH TAPE IS MOUNTED (CUU):281
ENTER DEVICE TYPE (2401,2415,2420,3420):2401
ENTER DEVICE ADDRESS WHERE SYSTEM RESIDENCE WILL BE BUILT (CUU):350
ENTER DEVICE TYPE (2319,2314,3330,2305):3330

***SYSTEM DEFINITION COMPLETED***

00E PRINTER
00D PUNCH
00C READER
280 PID TAPE
281 SCRATCH TAPE
350 NEW SYSTEM RESIDENCE
ARE THE ABOVE ENTRIES CORRECT (YES,NO):yes

VM/370 VERSION vv LEVEL 0 PLC 0000; mm/dd/yy hh:mm:ss
NOW 08:54:23 EDT FRIDAY mm/dd/yy
CHANGE TOD CLOCK (YES|NO):yes
SET DATE MM/DD/YY :mm/dd/yy
SET TIME HH:MM:SS :09:04:36
PRESS "TOD ENABLE SET" KEY AT DESIGNATED INSTANT

NOW 09:04:36 EDT FRIDAY mm/dd/yy
CHANGE TOD CLOCK (YES|NO):no
09:04:38 START ((COLD|WARM) (DRAIN))|(SHUTDOWN):cold
09:04:40
09:04:42 LOGON AT 09:04:37 EDT FRIDAY mm/dd/yy
LINE 01F LOGON AS CPGEN      USERS= 001
09:04:42

DMKCPI952I nnnnK SYSTEM STORAGE
```

If you have not defined your system residence volume with a label of VMREL2, two messages indicating that the volume labeled VMREL2 is not mounted are issued. If you have labeled it as VMREL2, a message indicating VMREL2 conflict is issued. You can ignore the following messages.

```
09:04:43 FILES:  NO RDR,  NO PRT,  NO PUN
09:04:44

DMKIOG556I FORMATTING I/O ERROR RECORDING MASK

09:04:46

DMKIOG557I FORMATTING MCH ERROR RECORDING AREA
```

The TOD clock referred to is the System/370 Time of Day clock. Enter the actual date and time in response to the "SET DATE" and "SET TIME" messages, and press the TOD Enable Set switch on the system control panel when the exact time specified agrees with the installation wall clock.

If, for some reason, you must reload the starter system, the following message is displayed:

```
VM/370 STARTER SYSTEM VERSION 2.0

DO YOU WISH TO RE-DEFINE YOUR SYSTEM (YES|NO):
```

Respond by entering "no", unless the failure was the result of incorrectly specifying the device addresses, or unless the tapes and/or disks have been moved. If you reply "yes", you repeat Step 7.

STEP 8. SET THE TERMINAL MODE AND SPOOL THE CONSOLE

If the system console you are using is a display device, you should at this point, spool your console output so that you have a record of what you do. To spool the console input and output, issue the command:

```
spool console start
```

to save a copy of the system generation.

Because the default terminal environment for the primary system operator is CP, you should also issue the command:

```
terminal mode vm
```

The virtual machine terminal mode lets you remain in the CMS environment when you enter data on the display device.

STEP 9. DEFINE OR ATTACH THE SYSTEM RESIDENCE DEVICE

The CPGEN virtual machine assumes that the system residence volume is labeled VMREL2 and is at virtual address 350 if the device type is 3330 or at virtual address 351 if the device type is 2314. If you labeled your system residence volume VMREL2 in Step 2 and designated 350 at its virtual address in Step 7, your system residence device is already available.

If you did not label your system residence device VMREL2, you must now attach it to your virtual machine, CPGEN. Use Procedure 2 to attach your system residence devices.

For example, if you designated the 131 drive, labeled VMREL2, as your system residence device for the generation procedure in Step 7, now you

must define your virtual device 350 so that it corresponds to the real disk 131.

Use one of the following procedures to define or attach your system residence volume:

- Procedure 1

If you labeled your system residence volume VMREL2, but did not define its address as 350, you must do so now. Press the Request key and enter the command:

```
define 350 as 131
```

You, as the operator of the CPGEN virtual machine, gain ownership of that entire volume as virtual address 350. The CP DEFINE command maps this virtual address to the real address of your residence volume as defined in your DMKSYS module.

Remember that you restored your starter system tape to a 3330 device. If you are now creating a 2314 system residence, with label VMREL2, issue the command:

```
define 351 as 131
```

You cannot create a 3340 system residence directly from a 3330 starter system. However, a 3340 starter system is provided and a procedure for generating a 3340 system residence volume from a current 2314 or 3330 system residence volume and a PLC tape is described in Appendix H. VM/370 recommends that you use the 3340 starter system.

- Procedure 2

If you did not label your system residence volume as VMREL2, you must now attach your system residence volume to your virtual machine. Press the Request key and enter the command:

```
attach 350 to cpgen as 350
```

Note: The first device address you specify is for the real device; the second device address is for the virtual device. The real 350 is the system residence volume you formatted in Step 2. The virtual 350 is the system residence device defined in DMKSYS (with the SYSRES macro) and also entered in response to the message:

```
ENTER DEVICE ADDRESS WHERE SYSTEM RESIDENCE WILL BE BUILT (CUU):
```

STEP 10. MAKE OTHER DEVICES AVAILABLE

You must attach your tape drives and designate the printer to receive abnormal termination dumps if they occur. Press the Request key to enter each of the following commands:

```
attach 280 to cpgen as 181
attach 281 to cpgen as 182
set dump 00e
```

In the first command, the real tape drive (280) attached must be the same real address you entered in Step 7 in response to the message:

```
ENTER ADDRESS WHERE PID TAPE IS MOUNTED (CUU):
```

This real device must be attached to CPGEN as 181, because the VMFBLD EXEC procedure (which is used later) expects it to be 181.

In the second command, the real tape drive (281) attached must be the same real address you entered in Step 7 in response to the message:

```
ENTER ADDRESS WHERE SCRATCH TAPE IS MOUNTED (CUU):
```

This real device must be attached to CPGEN as 182, because the GENERATE EXEC procedure (which is used later) expects it to be 182.

If only one tape drive is available, enter

```
attach 280 to cpgen as 181
```

You can define your virtual 181 as 182 later in the system generation procedure.

The address of the real printer, defined in Step 7, is 00E. Any system dumps that occur are directed to that address.

If you wish, you can issue the CP command:

```
query virtual all
```

before and after performing Step 10, to see how your virtual machine's configuration changes.

STEP 11. LOAD CMS

Next, you must load CMS from virtual address 190 by issuing the CP command:

```
ipl 190
```

After CMS is loaded, a message is displayed on the system console indicating that CMS has successfully loaded:

```
CMS VERSION 2.0 mm/dd/yy hh.mm
```

Press the Return key and the CPGEN 191 minidisk is accessed as your A-disk.

STEP 12. LOAD THE PLC (PROGRAM LEVEL CHANGE) TAPE

Demount the PID (Program Information Department) starter system tape, and mount and ready the PLC tape at 280.

The PLC tape is prepared by the VM/370 Program Level Change (PLC) service. This is a system update service that can include new functions as well as cumulative system changes. The latest PLC tape contains all new updates, as well as all previous updates since the last VM/370 release base. IBM Field Engineering is responsible for initially ordering the PLC service. Thereafter, PID automatically ships the PLC tapes to the FE location, and FE is responsible for applying the updates to VM/370 systems.

The PLC tape supplied with the starter system should be installed as part of the system generation procedure. Issue the CMS command:

```
tape load
```

The system issues the following messages:

```
LOADING.....

VMFBLD   EXEC     A1
END-OF-FILE OR END-OF-TAPE
R;
```

The VMFBLD EXEC procedure is a service program that adds or replaces CP, CMS, or RSCS object modules. It loads the updates from the PLC tape for you. Issue the command:

```
access 191 c
```

because VMFBLD accesses other disks as its A-disk during processing. Next, issue the command:

```
vmfbld cp
```

to execute VMFBLD.

The VMFBLD EXEC prompts you with the message:

```
ENTER CP STAGING AREA DISK ADDRESS:
194
```

You respond with 194. The 194 minidisk belongs to the CPGEN virtual machine and is the area on the restored starter system volume that contains all the object modules necessary to generate a CP nucleus. The contents of the minidisk are updated with the latest versions of all object modules, as well as the latest version of the DMKMAC macro library. The following message is displayed:

```
ARE YOU A NEW USER? -- RESPOND (YES|NO)
```

You must respond "yes" to this question. The following messages are then displayed:

```
'190' REPLACES 'A(194)'
190 ALSO = S-DISK
hh:mm:ss REWIND COMPLETE
CMS VERSION v.0 - mm/dd/yy hh.mm
```

The VMFBLD EXEC procedure links to the CMS system disk and accesses it as the A-disk. Next, VMFBLD reads the updated service programs from the PLC tape and writes them to the CMS S-disk. Then VMFBLD loads the new CMS system onto the system disk. After CMS is loaded, a message is displayed to indicate that the CMS system was successfully loaded:

```
CMS VERSION v.0 - mm/dd/yy hh.mm
```

You must enter a null line to continue.

STEP 13. PUNCH THE SERVICE PROGRAMS

Use the GENERATE EXEC procedure to punch the service programs. Issue the command:

```
generate srvcpgm
```

These service programs are needed for standalone use; you shculd externally identify the decks and keep them intact.

The programs punched are:

- DMKFMT (A three-card loader precedes the DMKFMT text deck)
- DMKDIR (A three-card loader precedes the DMKDIR text deck)
- DMKDDR (A three-card loader precedes the DMKDDR text deck)
- IBCDASDI (The IBCDASDI text deck is loadable)

The GENERATE EXEC issues the following message:

```
THE FOLLOWING STANDALONE SERVICE PROGRAMS ARE BEING PUNCHED
      ** FORMAT - DIRECT - DUMP/RESTORE - IBCDASDI **

PUNCHING ' IPL FMT ' ******

PUNCHING ' IPL DIR ' ******

PUNCHING ' IPL DDR ' ******

PUNCHING ' IPL IBCDASDI ' ******
```

Each program deck is preceded by a CP userid card and several separator cards, all of which may be discarded. The format of these cards is described in the VM/370: Operator's Guide.

For more information about the GENERATE EXEC procedure, see "Part 6: Updating VM/370."

STEP 14. PRINT AND PUNCH THE STARTER SYSTEM SUPPLIED DIRECTORY, DMKSNT, DMKSYS, AND DMKFCB

After the service programs are punched, the GENERATE EXEC asks you whether you want a copy of the Release 2 directory printed. You shculd respond "yes."

```
PRINT COPY OF RELEASE2 DIRECT? -- RESPOND (YES|NO): yes
```

This prints a copy of the Release 2 directory, as well as copies of the DMKSYS ASSEMBLE (the system definition file), DMKFCB ASSEMBLE (the forms control buffer file), and DMKSNT ASSEMBLE (the system name table) provided with the starter system.

The GENERATE EXEC procedure issues the following message:

```
A SAMPLE DIRECTORY IS BEING PRINTED TO AID YOU.
IT SHOWS WHERE THE VIRTUAL DISKS ARE LOCATED ON 'CPV2L0'
YOU MAY USE THESE MINIDISKS FOR OTHER VIRTUAL MACHINES,
IN PARTICULAR THE CMS SYSTEM DISK ( MAINT 190 ) AND
THE CP STAGING AREA DISK ( MAINT 194 )
INCLUDED IN THIS DIRECTORY IS THE USERID: MAINT
WHICH WILL BE USED FOR FUTURE SUPPORT OF THE SYSTEM.
```

THIS USERID SHOULD BE INCLUDED IN THE DIRECTORY
YOU BUILD FOR YOUR FLOOR USE.
** CAUTION ** IF YOU DESTROY USER MAINT'S AREAS, IT WILL BE
NECESSARY TO RE-BUILD THE ENTIRE SYSTEM.

A SAMPLE OF DMKSYS, DMKFCB, AND DMKSNT ASSEMBLE ARE ALSO BEING
PRINTED TO AID YOU. THIS SAMPLE DMKSNT IS BASED ON THE
INFORMATION INCLUDED IN THE SAMPLE DMKSYS AS WELL AS THE
EXAMPLE ALLOCATIONS FOR VMREL2 PROVIDED IN THE SYSGEN GUIDE.
A COPY OF THIS DMKSNT MODULE HAS BEEN INCLUDED IN THE CP NUCLEUS,
SUCH THAT IF ONE USES THE INCLUDED DMKSYS AND THE

SAMPLE ALLOCATION PROVIDED IN THE SYSTEM GENERATION GUIDE, HE WILL BE ABLE TO SAVE HIS CMS SYSTEM UPON COMPLETION OF THE SYSTEM GENERATION PROCEDURE. A COPY OF DMKFCB HAS BEEN INCLUDED IN THE NUCLEUS AND NEED NOT BE RE-ASSEMBLED FOR SYSTEM GENERATION. IT HAS BEEN INCLUDED FOR THE USER WHO WOULD LIKE TO MODIFY OR ADD TO THE EXISTING BUFFER LOAD.

NOTE: IF THE USER WISHES TO MODIFY THE SAMPLE DMKSNT AND/OR DMKFCB HE MAY INCLUDE THE UPDATED SOURCE WITH THE SOURCE INCLUDED UNDER THE OPTION 'GENERATE VM370', OF THE SYSTEM GENERATION PROCEDURE. IF PRESENT, IT WILL AUTOMATICALLY BE ASSEMBLED AND INCLUDED IN THE NEW CP NUCLEUS.

Again, the GENERATE EXEC procedure prompts you:

DO YOU WISH TO HAVE A COPY OF DMKSNT, DMKSYS, DMKFCB, AND RELEASE2 DIRECT PUNCHED TO CARDS? -- RESPOND (YES|NO):

You should enter "yes" to have these decks punched. "Part 2: Defining Your VM/370 System" contains a listing of the Release 2 VM/370 directory and the DMKSNT, DMKSYS, and DMKFCB modules supplied with the starter system. The following messages are issued to indicate the files that are being punched:

```
PUNCHING ' DMKSNT ASSEMBLE ' ******

PUNCHING ' DMKSYS ASSEMBLE ' ******

PUNCHING ' DMKFCB ASSEMBLE ' ******

PUNCHING ' RELEASE2 DIRECT ' ******
R;
```

STEP 15. BUILD THE VM/370 DIRECTORY AND ASSEMBLE THE FILES DEFINING THE REAL I/O AND SYSTEM DEVICES

Next, you should place the files describing your installation's version of the VM/370 directory, real I/O configuration, and system devices in the reader and invoke the GENERATE EXEC procedure to build the directory and assemble the files describing the configuration. Place the following cards in the card reader, in the sequence shown:

```
ID CPGEN
:READ  filename DIRECT
(Directory program control statements)
:READ  DMKRIO ASSEMBLE
(real I/C configuration macros)
:READ  DMKSYS ASSEMBLE
(CP system control file macros)
```

The Directory program control statements, real I/O configuration macros, and CP system control file macros are those you created according to the instructions in "Part 2: Defining Your VM/370 System."

Also, if you want, you can change the printer buffer load (DMKFCB) or the named system (DMKSNT) modules to conform to local requirements. If you want to change the printer buffer load or the system name table, you should add your changes to the files already punched (in Step 14) and place the following files in the card reader behind the CP system control file macros:

```
:READ DMKFCB ASSEMBLE
(forms control macros)
:READ DMKSNT ASSEMBLE
(system name table macros)
```

Instructions for creating new forms control macros are in the VM/370: System Programmer's Guide. The system name table macros are described in "Part 2: Defining Your VM/370 System" of this manual.

FORMAT OF THE READ CONTROL STATEMENT

The READ control statements must be punched according to the following format.

Column	Number of Characters	Contents	Meaning
1	1	colon':'	Identifies card as a control card
2-5	4	READ	Identifies card as a READ control card
6-7	2	blank	
8-15	8	fname	Filename of the file punched
16	1	blank	
17-24	8	ftype	Filetype of the file punched
25-80	56	blank	

SPECIAL PROCEDURE IF YOU ARE USING ONLY ONE TAPE DRIVE

If you are using only one tape drive, at this time you must issue the command:

```
define 181 as 182
```

Now mount the scratch tape in place of the PLC tape and ready the device.

INVOKE THE GENERATE EXEC PROCEDURE

When all the files are placed in the reader, invoke the GENERATE EXEC procedure by issuing the following command:

```
generate vm370
```

This procedure invokes the Directory service program to build the disk-resident VM/370 directory, then assembles the DMKRIC and DMKSYS files that you placed in the real card reader. After the Directory service program execution completes, the DMKRIO and DMKSYS files are

assembled in preparation for building the new CP system nucleus. GENERATE then checks for DMKFCB and DMKSNT source files. When new versions of the DMKFCB and DMKSNT modules are provided, GENERATE assembles the new modules and replaces the corresponding modules supplied with the starter system with the new modules. If any errors occur while the VM/370 directory is being built, the Directory program issues error messages and the GENERATE EXEC procedure issues the following message:

```
CORRECT THE DIRECTORY CARDS AND RELOAD THE CARD READER
  RESPOND WITH: GENERATE DIRECT
```

Correct the errors in the Directory program control statements, and reload the card reader with only the ID card, the :READ statement, and the Directory program control statements. Then respond with:

```
generate direct
```

If errors are detected while the DMKRIO, DMKSYS, DMKFCB, or DMKSNT files are assembling, GENERATE issues a similar message:

```
CORRECT THE filename ASSEMBLE FILE AND RELOAD THE CARD READER
  RESPOND WITH: GENERATE filename
```

You must correct the errors in the indicated file, and reload the card reader with only the ID card, the :READ statement, and the appropriate file. Then, issue the GENERATE command with the appropriate option. For example:

```
generate dmksys
```

STEP 16. SPECIAL CONSIDERATIONS FOR VIRTUAL=REAL MACHINES

Once the directory is built and the files are assembled, the GENERATE EXEC procedure asks if you want the virtual=real option included in your CP nucleus:

```
VIRTUAL=REAL OPTION REQUIRED (YES,NO):
```

If you respond "yes" to this question, you are prompted to enter the amount of storage you wish to reserve in the CP nucleus for a virtual=real machine. ("Part 1. Planning for System Generation" contains formulas to help you determine how much storage you need to reserve.) This size must be a multiple of 4K and must be entered in the format nnnnK. The minimum size you can specify is 32K and the maximum is 4M. For example:

```
STORAGE SIZE OF VIRT=REAL (MINIMUM IS 32K):
100k
0100K STORAGE SIZE FOR VIRTUAL=REAL
IS THE ABOVE ENTRY CORRECT (YES,NO):
yes
```

STEP 17. LOAD THE CP NUCLEUS

Once you respond to the virtual=real generation questions, the GENERATE EXEC procedure writes the CP nucleus on tape. To do this, GENERATE first issues the VMFLOAD command to punch the loader and CP object modules to a virtual punch spool file. Then it transfers this file to

the virtual card reader file and writes the file on tape. GENERATE issues the following messages during the processing:

```
hh:mm:ss NO FILES PURGED
SYSTEM LOAD DECK COMPLETE
hh:mm:ss PUN FILE nnnn TO CPGEN COPY 01 NOHOLD
IPLABLE NUCLEUS NOW ON TAPE ****
hh:mm:ss PRT cuu OUTPUT OF CPGEN FILE=nnnn RECDS=nnnnnn
  COPY = 01 A PRT
```

Note: If any errors are detected while the tape is being written, you must recreate the CP nucleus. To do this, issue the command:

```
generate cp nucleus
```

The procedure restarts at Step 16 where you are asked if you want the virtual=real option.

LOADING A CP NUCLEUS WITHOUT A VIRTUAL=REAL AREA

If you respond "no" when asked if you wanted the virtual=real area, the GENERATE EXEC procedure loads the nucleus for you. You receive the following message:

```
WHEN 'NUCLEUS LOADED ON XXXXXX' IS TYPED, ISSUE 'CLOSE PRT',
  TO GET THE CPLOAD MAP. WHEN PRINTING IS COMPLETE,
  SHUTDOWN THE SYSTEM AND IPL THE NEW SYSRES VOLUME.
```

When the nucleus is written on the system residence volume, the message

```
NUCLEUS LOADED ON volid
```

is issued, where volid is the volume serial number of your system residence volume. The volid is the serial number you specified on the SYSRES macro when you prepared the CP system control file (DMKSYS). If you followed the example in this manual, the serial number of your system residence volume is VMREL2.

To print the load map you must close the printer; issue the command:

```
close prt
```

When printing is complete, shutdown the system in preparation for loading your newly generated CP nucleus:

```
shutdown
```

LOADING A CP NUCLEUS THAT HAS A VIRTUAL=REAL AREA

If you responded "yes" when asked if you wanted the virtual=real area, the GENERATE EXEC procedure issues the following message:

```
TO LOAD THE CP NUCLEUS JUST CREATED, SHUTDOWN THE SYSTEM AND
THEN IPL THE TAPE. THE CPLOAD MAP WILL AUTOMATICALLY BE PRINTED AT THE
PRINTER WHOSE ADDRESS IS '00E'. IF THERE IS NO PRINTER AT THIS ADDRESS
THE LOAD MAP WILL BE PRINTED AT THE FIRST PRINTER CAUSING AN INTERRUPT
(IE. NOT-READY TO READY SEQUENCE). ONCE THE NUCLEUS HAS BEEN LOADED,
YOU MAY IPL YOUR NEW CP SYSTEM RESIDENCE VOLUME.
```

NOTE: THERE MUST BE ENOUGH STORAGE ON THE SYSTEM (VIRTUAL OR REAL), TO CONTAIN THE VIRT=REAL AREA AND THE CP NUCLEUS.

You must be sure that there is enough storage to load the CP nucleus with a virtual=real area before you load it. The "Specifying a Virtual=Real Machine" section of Part 1 tells you how to determine the amount of real or virtual storage you need. You can load a CP nucleus that has a virtual=real area in either a real or virtual machine.

You IPL the tape containing the CP nucleus next. The CP nucleus is on the tape at virtual address 182 for the CPGEN virtual machine.

STEP 18. IPL THE NEWLY GENERATED VM/370

IPL the newly generated system residence volume by setting the load unit address dials to the address of the system residence volume and pressing the Load pushbutton. The userid OPERATOR (or whatever userid you specified as the system operator on the SYSOPER macro) is logged on when you IPL. The following messages are issued. Respond as shown.

```
VM/370 VERSION vv LEVEL 00 PLC nnnn; mm/dd/yy hh:mm:ss

NOW hh:mm:ss EDT day mm/dd/yy
CHANGE TOD CLOCK (YES|NO):no
hh:mm:ss START ((COLD|WARM) (DRAIN))|(SHUTDOWN): cold
hh:mm:ss LOGON AT hh:mm:ss EDT day mm/dd/yy
hh:mm:ss LINE 01F LOGON  AS OPERATOR USERS = 001
hh:mm:ss

DMKCPI952I nnnnK SYSTEM STORAGE

hh:mm:ss FILES:  NO RDR,  NO PRT,  NO PUN
hh:mm:ss

DMKIOG556I FORMATTING I/O ERROR RECORDING AREA

hh:mm:ss

DMKIOG557I FORMATTING MCH ERROR RECORDING AREA
```

At the time you IPL your newly generated VM/370, your system residence is formatted according to your specification on the SYSRES macro. Also, if you used the VM/370 directory supplied with the 3330 starter system, your starter system volume (CPV2L0) is set up as shown in Figure 20. Be careful not to modify the 190 and 194 minidisks for the MAINT user. These minidisks, and the information they contain, are required for applying PLC (Program Level Change) updates to VM/370.

Real Cylinder	Number of Cylinders	Contents
0	1	Unused
1	1	191 minidisk for the IVPM1 user
2	1	191 minidisk for the IVPM2 user
3	1	191 minidisk for the REM2780 user
4–10	7	191 minidisk for the OPERATOR user
11–15	5	191 minidisk for the CE user
16–22	7	191 minidisk for the MAINT user
23–28	6	191 minidisk for the ECMODE user
29	1	199 minidisk for the MAINT user
30–105	76	190 minidisk for the MAINT user Note: The last cylinder of this minidisk contains the CMS nucleus.
106–149	44	194 minidisk for the MAINT user
150–403	254	Not used.

Figure 20. Allocation of the Starter System Volume (CPV2L0) When the 3330 Starter System Directory Is Used

STEP 19. BACK UP THE NEWLY GENERATED VM/370

At this time, you should be sure to back up your new system residence volume. If your real machine has at least 448K bytes of real storage, the tape created in Step 17 is sufficient backup. However, that tape is not sufficent backup for real systems with less than 448K of real storage because that tape cannot be loaded on such systems.

VM/370 systems that execute on a real machine with less than 448K bytes of real storage should use the DASD Dump Restore (DDR) service program to create a backup tape similar to the one created in Step 17. The DASD dump restore program is described in Appendix F. If your system residence volume is at address 350 and you labeled it VMREL2, you could use the following DDR control statements to back it up:

```
input 350 3330 vmrel2
output 180 3420
dump cpvol
```

The tape produced contains the CP nucleus; it does not contain the VM/370 directory. However, you already created a standalone version of the Directory program in Step 13 and have the Directory program control statements, which you used in Step 15, to recreate the VM/370 directory.

If you do not wish to use the DDR program to back up your system, you can load the tape produced in Step 17 in a virtual machine. If you load the tape in a virtual machine, that virtual machine must have (1) 512K

of storage and (2) write-access to the system residence volume, at the address defined for system residence in the SYSRES macro of the CP System Control (DMKSYS) file.

When you use the DDR program to back up your system you do not get a load map when you restore the tape. You do get a load map if you load the tape produced in Step 17.

STEP 20. MOVE THE CMS SYSTEM DISK, IF YOU WISH

The CP nucleus just created is tailored to your installation's exact I/O configuration with all the appropriate options selected. The CMS nucleus requires no special tailoring for installation. The CMS system residence disk was created when the VM/370 starter system was loaded; it contains a copy of the CMS nucleus and disk-resident command modules. At this point, the CMS system resides on the virtual 190 disk for the user, MAINT. The operator's VM/370 directory has an entry that links to MAINT's 190 disk, therefore you can invoke CMS by issuing the command:

```
ipl 190
```

and thus have access to most of the CMS functions. Although most of the CMS functions are available, CMS has not yet been updated with the latest PLC changes. The Assembler that is available now is the one supplied with the starter system.

If you wish, you can move the CMS system disk to a disk other than the one designated by the starter system. To move the CMS system disk to another minidisk, use the COPY function of the DDR service program. The new minidisk should be the same size as the CMS system disk, but it does not need to be formatted. Also, if you move the CMS system, you must make the appropriate changes to the userid MAINT in your VM/370 directory (that is, change the MDISK entry for 190).

If you add IBM Program Products or your own disk-resident modules to the CMS system disk, you may have to create a new system disk to contain the additional modules. The "Creating a CMS System Disk" section of "Part 6. Updating VM/370" describes the procedure.

STEP 21. FORMAT THE OPERATOR'S VIRTUAL 191 DISK

Before any new minidisk area can be used for CMS files, it must be initialized with the CMS FORMAT command, which formats the area into 800-byte blocks. Take care not to format areas which contain data restored from the starter system (such as the 190 and 194 minidisks belonging to the user MAINT.) The CMS FORMAT command is described in the VM/370: Command Language Guide for General Users.

Note: After you complete this step, a portion of the starter system is overlaid by the operator's virtual 191 minidisk. If for any reason you wish to IPL the starter system again, you must restart at step 1.

At this time you are logged on as the operator. Use the following procedure to format your virtual disk 191. First, load CMS (IPL 190). CMS responds with:

```
CMS VERSION 2.0 - mm/dd/yy hh:mm
```

Next, enter the following command:

```
access (nodisk
```

The NODISK option prevents CMS from automatically accessing your virtual disk 191. (Accessing 191 at this time would cause an error message to be issued, because 191 is not yet initialized, and therefore cannot be used.) After the Ready message is displayed, issue the command:

```
format 191 a
```

The CMS FORMAT command prompts you with the following message:

```
FORMAT WILL ERASE ALL FILES ON DISK 'A(191)'. DO YOU WISH TO
  CONTINUE? (YES|NO):
```

If you respond yes, CMS prompts you with:

```
ENTER DISK LABEL:
opr191
```

Enter the one- to six-character alphameric label of the virtual disk. You can use whatever label you wish for this virtual disk. In this example, the label is OPR191. CMS then issues:

```
FORMATTING DISK 'A'.
'07' CYLINDERS FORMATTED ON 'A(191)'.
```

and a Ready message.

STEP 22. FORMAT THE MAINT USER'S VIRTUAL 191 DISK

Next, you should initialize the 191 disk belonging to MAINT, in the same way you initialized the operator's 191. Log yourself off the system and log on again as userid MAINT, using the passwcrd CPCMS. Now IPL 190. When the CMS Ready message is displayed, issue

```
access (nodisk
```

and continue to format MAINT's 191, using the same procedure you used to format OPR191.

STEP 23. UPDATE CMS

Now that the MAINT 191 minidisk is formatted, you (as the MAINT user) can update CMS with the changes supplied on the PLC tape. Mount the PLC tape, if it is not already mounted. You must attach the real tape drive to your CMS virtual machine (MAINT); for example:

```
attach 280 to maint as 181
```

and, with the 191 minidisk that belongs to MAINT accessed as your A-disk, rewind and load the tape:

```
tape rew
tape load
```

CMS responds with the following message:

```
LOADING.....
VMFBLD   EXEC   A
END-OF-FILE OR END-OF-TAPE
```

BUILDING A NEW CMS

Now execute the VMFBLD EXEC procedure by issuing the commands:

```
access 191 c
vmfbld cms
```

The VMFBLD EXEC procedure prompts you with:

```
ENTER CMS SYSTEM DISK ADDRESS:
```

You respond with "190". Next, VMFBLD issues the message:

```
190 ALSO = S-DISK.
```

You are asked if you want the service programs punched:

```
PUNCH STANDALONE SERVICE PROGRAMS--RESPOND (YES|NO):
```

The following messages are displayed; respond as shown:

```
hh:mm:ss  NO  FILES PURGED
hh:mm:ss  NO  FILES PURGED
SYSTEM LOAD DECK COMPLETE
hh:mm:ss PUN FILE nnnn  TO  CPGEN    COPY 01 NOHOLD
hh:mm:ss REWIND COMPLETE
         WHEN THE NEW CMS SYSTEM IS BUILT ISSUE THE FOLLOWING
         COMMAND
              CLOSE PRT . . . (PRINTS THE LOAD MAP)

DMSINI606R SYSTEM DISK ADDRESS = 190
DMSINI615R Y-DISK ADDRESS = 19e
DMSINI607R REWRITE THE NUCLEUS ? yes
DMSINI608R IPL DEVICE ADDRESS = 190
DMSINI609R NUCLEUS CYL ADDRESS = 75
DMSINI610R ALSO IPL CYLINDER 0 ? yes
DMSINI611R VERSION IDENTIFICATION =
DMSINI612R INSTALLATION HEADING =
CMS VERSION 2.0 - mm/dd/yy hh:mm
```

In the example, the 190 you entered is the 190 minidisk that belongs to the user MAINT; it is equivalent to the 190 minidisk that belongs to CPGEN on the starter system. If you used the VM/370 directory supplied with the starter system, you allocated 76 cylinders (virtual cylinders 0-75) to the 190 minidisk belonging to the user MAINT. The last cylinder of this minidisk contains the CMS nucleus, therefore you specify 75 as the nucleus cylinder address when responding to message DMSINI609R.

When the VMFBLD EXEC procedure completes its execution, the CMS system residence volume is updated with the most current object modules (text decks) and load modules, and the new CMS nucleus is written on the CMS system residence volume.

Also, if there are any updates for the EREP program or the Assembler on the PLC tape, the VMFBLD EXEC procedure updates those programs and creates the corresponding auxiliary directories.

STEP 24. SAVE CMS

If you used the sample DMKSNT and DMKSYS supplied with the starter system, and used the sample allocations shown in Step 2, you can now save your CMS system.

To save the CMS system, you must load it and then save it as soon after loading as possible. Issue the command:

```
ipl 190
```

When the terminal unlocks, do not press carriage return, but immediately issue the command:

```
savesys cms
```

Then press carriage return. If CMS is successfully saved, the message

```
hh:mm:ss SYSTEM SAVED
CMS VERSION 2.0 - mm/dd/yy hh:mm
```

is displayed. Your CMS system is now saved; you can issue IPL CMS instead of IPL 190, when you wish to run CMS.

If you named your CMS system something other than CMS, such as CMS1, the entry you made in the system name table would be for CMS1, you would have to save CMS1 (SAVESYS CMS1) and then you could IPL CMS1. For more information about how to save CMS, see the VM/370: System Programmer's Guide or the "Saved Systems" section of this manual.

At this point, use the Installation Verification Procedure (IVP) to test the new system. Log off as the userid MAINT and log on again as the userid OPERATOR, using the password OPERATOR. The IVP is described later in Part 3.

Generating CP and CMS Using the 3340 Starter System

Except where otherwise noted, you can, if you wish, substitute other values in place of the device addresses, volume labels, and allocations shown. Note that if you use the sample DMKSNT and DMKSYS provided with the starter system, and use the sample allocations shown in Steps 2 and 3, you can save your CMS system at the end of the procedure.

The 3340 Model 35 and 3340 Model 70 starter systems are identical. If you create a 3340 Model 70 starter system, cylinders 349-697 are not used.

It is strongly recommended that you use the sample allocations given in Step 2 and the label VMREL2 for the new system residence volume, to ensure that you have sufficient TEMP space to complete the system generation. (The TEMP space provided on the starter system volume may not be sufficient for large systems.)

The examples of messages and responses assume that you are performing the system generation at a typewriter terminal, such as a 3210, 2741, or 3767 (operating as a 2741). If you are using a display device, such as the 3277, when you type the response to a prompting message that response appears in the user input area. When you enter that response, it is redisplayed in the output area on the line below the prompting message. Also, if the standalone service programs (such as the DASD Dump Restore program or Format/Allocate program) send output to a terminal display screen, the output is wrapped around immediately, when the screen becomes full, to continue displaying.

While you are generating the system, you may see some extraneous messages as the starter system does its processing. These are not shown in the examples below. Only those messages that you should take note of or respond to are shown.

STEP 1. LOAD THE FORMAT PROGRAM FROM THE STARTER SYSTEM TAPE

Mount the CP starter system tape and IPL the tape. The CP Format/Allocate service program is the first file on the tape; it is now loaded. Do not rewind the tape because the next file is needed later in the system generation procedure (Step 4).

STEP 2. FORMAT, LABEL, AND ALLOCATE THE SYSTEM RESIDENCE VOLUME

Use the CP Format/Allocate program to format, label, and allocate space on the new system residence volume. First, identify the system console by pressing the Request key (or equivalent); if the console address is either 009 or 01F, you do not have to press the Request key. Then, to execute the Format/Allocate service program, respond to the prompting messages.

In the following example, the responses (format, 341, 3340-35, 000, 348 and VMREL2) format the 3340 disk at address 341 and label it VMREL2. The values you should specify for a 3340 model 70 are shcwn in parentheses. Whatever label you specify must match what you specified on the SYSVCL operand of the SYSRES macro when you defined the system.

In any case, do not use CPV2L0 because that is the label of the starter system disk. The console output looks like:

```
VM/370 FORMAT/ALLOCATE PROGRAM VERSION 2.0

ENTER FORMAT OR ALLOCATE:format
FORMAT FUNCTION SELECTED
ENTER DEVICE ADDRESS (CCU):341
ENTER DEVICE TYPE:3340-35    (or, 3340-70)
ENTER START CYLINDER (XXX) OR "LABEL":000
ENTER END CYLINDER (XXX):348    (or, 697)
ENTER DEVICE LABEL:vmrel2
FORMAT STARTED
FORMAT DONE
000 PAGE RECORDS FLAGGED
```

When the Format/Allocate operation completes, the prompting message:

```
ENTER FORMAT OR ALLOCATE:
```

is displayed. Now that the system residence volume is formatted and labeled, you must allocate the disk space. Again, you must respond to the prompting messages. In the following example, the space on the 3340 disk at address 341, with the label VMREL2, is allocated (11 cylinders of permanent space, 5 cylinders of directory, 310 cylinders of temporary space, and 23 cylinders of TDSK space). You can use the formulas given in the "Creating Your VM/370 Directory" section of Part 2 tc ensure that five cylinders are enough space for your VM/370 directory. If you do not allocate your DASD space as shown in this example, you are responsible for ensuring that you have enough temporary space to perform the assemblies associated with VM/370 system generation.

```
ENTER FORMAT OR ALLOCATE:allocate
ALLOCATE FUNCTION SELECTED
ENTER DEVICE ADDRESS(CCU):341
ENTER DEVICE TYPE:3340-35    (or, 3340-70)
ENTER DEVICE LABEL:vmrel2
ENTER ALLOCATION DATA FOR VOLUME VMREL2
TYPE CYL CYL
.... ... ...
perm 000 010
drct 011 015
temp 016 325
tdsk 326 348
end
ALLOCATION RESULTS
PERM 000 010
DRCT 011 015
TEMP 016 325
TDSK 326 348
DEVICE 341 VOLUME VMREL2 ALLOCATION ENDED
```

If your system residence volume is a 3340 Model 70, the allocation results messages show the unused cylinders (349-697) as temporary space.

Note: If you encounter I/O errors on alternate tracks, allocate the cylinder as permanent (PERM) space.

STEP 3. LABEL THE STARTER SYSTEM VOLUME

Use the Format/Allocate program to label the scratch volume that is to contain the CP starter system. This label must be CPV2L0. You can format and label this volume, or just label it. Formatting is unnecessary because you are going to restore the starter system to this volume. In the following example, the responses (format, 340, 3340-35, label, and cpv210) to the prompting messages put the label CPV2L0 on the 3340 disk at address 340.

```
ENTER FORMAT OR ALLOCATE: format
FORMAT FUNCTION SELECTED
ENTER DEVICE ADDRESS (CCU):340
ENTER DEVICE TYPE:3340-35 (or, 3340-70)
ENTER START CYLINDER (XXX) OR "LABEL":label
ENTER DEVICE LABEL: cpv210
LABEL IS NOW CPV2L0
```

When the Format/Allocate program is complete, it responds:

```
ENTER FORMAT OR ALLOCATE:
```

You do not have to respond to this message.

Now the volume is available and ready for the data that is to be placed on it by the DASD Dump Restore service program (module DMKDDR). The DASD Dump Restore program is the second file on the starter system tape.

STEP 4. LOAD THE DASD DUMP RESTORE PROGRAM FROM THE STARTER SYSTEM TAPE

IPL the starter system tape a second time to load the DASD Dump Restore (DDR) program. It is the second file on the starter system tape. Again, do not rewind the tape because the next file is needed in Step 5.

STEP 5. RESTORE THE STARTER SYSTEM TO DISK

Respond to the DDR prompting messages to restore the starter system.

In the following example, the starter system is restored from the 2400 series tape drive at address 280 to the 3340 disk at address 340 (and with label CPV2L0). The console output is:

```
VM/370 DASD DUMP/RESTORE PROGRAM VERSION 2.0
ENTER CARD READER ADDRESS OR CONTROL STATEMENTS
ENTER: sysprint cuu (cuu=real printer address)
ENTER: input 280 2400 (See the "Special Considerations for 800 BPI Tapes"
       section, if you ordered your starter system on 800 bpi tape)
ENTER: output 340 3340-35 cpv210   (or, output 340 334C-70 cpv210)
ENTER: restore all
RESTORING CPV2L0
END OF RESTORE
ENTER: (null line -- END key on 3215 or Enter key on 3277)
END OF JOB
```

The DDR program restores the third file on the starter system tape to the disk labeled CPV2L0. The restored disk contains:

- A VM/370 directory (RELEASE2 DIRECT), as well as sample source for DMKSYS, DMKSNT, and DMKFCB
- A VM/370 starter system nucleus
- A complete CMS system residence volume
- A complete CP system comprised of macro libraries and text files

When the disk is restored, continue with the system generation procedure. The format of the restored disk is shown in Figure 21.

Real Cylinder	Number of Cylinders	Contents
0	1	VM/370 directory
1–2	2	191 minidisk for the IVPM1 user
3–4	2	191 minidisk for the IVPM2 user
5–6	2	191 minidisk for the REM2780 user
7–10	4	CP nucleus
11	1	Warm start data
12–13	2	I/O Error Recording area
14–45	32	Spooling and paging space
46–47	2	191 minidisk for the CPGEN user
48–250	203	190 minidisk (the CMS system disk) for the CMSSYS user Note: The last two cylinders of the 190 minidisk (249–250) contain the CMS nucleus.
251–348	98	194 minidisk of the CPGEN user – it contains the CP object modules (text decks)

Note: Cylinders 349–697 are not used when the starter system is restored to a 3340 Model 70.

Figure 21. Format of a 3340 Restored Disk

Figure 21 represents the allocation of areas on a 3340 starter system volume.

CPGEN, CMSSYS, IVPM1, IVPM2, and REM2780 are user identifications that own certain minidisks on the starter system volume.

CPGEN is the userid of the operator (that is, you use this userid to control the real system and to build a version of CP tailored to your installation.

CMSSYS is a directory entry on the starter system volume that owns CMSSYS 190 (the CMS system disk). CPGEN has a read-only link to CMSSYS 190 in order to use it to create the new system. Using this link, the CPGEN user can read from, but not write on, the 190 minidisk belonging to the CMSSYS user.

IVPM1 and IVPM2 are used with the Installation Verification Procedure to test the new system.

REM2780 is used to test the 2780 as a remote entry station.

SPECIAL CONSIDERATIONS FOR 800 BPI TAPES

If you ordered the starter system on 800 bpi tape, you have two tapes. If you have two tape drives available for the system generation procedure, use the following form of the INPUT statement:

```
input 280 2400 281
```

In this case, when DDR finishes reading the first tape volume, it issues the message:

```
END OF VOLUME CYL xxx HD xx, MOUNT NEXT TAPE
```

and continues by reading the tape mounted on the alternate tape drive (181). If you have only one tape drive available, you enter an INPUT statement in the form:

```
input 280 2400
```

When the end-of-volume message appears, you must demount the first tape, mount the second tape on the same device and ready the device. Then, DASD Dump Restore program continues by reading the second tape.

STEP 6. IPL THE STARTER SYSTEM

You now load (IPL) the starter system from the disk you restored it to. In our example, you press the Load button with the load unit address dials set to 340. At this point, only the device containing the system residence volume, 340, is known to the starter system.

STEP 7. DEFINE THE DEVICES NEEDED TO DO THE SYSTEM GENERATION

If your system console is at an address other than 009 or 01F, after you load the starter system you must press the Request key (or equivalent key) to enable the starter system to recognize the system console. If the console is not recognized, the VM/370 starter system enters a wait state.

At this point both the system residence volume and system console are recognized by the starter system and you can define the other devices you need. The starter system supports up to 16 channels, 8 control units, and 16 devices. The real control blocks for these devices are not built in the standard manner; the starter system builds them dynamically.

The starter system program (DMKSSP) then prompts you to answer the following questions until all the real contrcl blocks necessary to operate a minimum machine configuration are created. The following example assumes: a 1403 printer at address 00E, a 2540 card reader/punch at addresses 00C (reader) and 00D (punch), tape drives at addresses 280

| and 281, and a 3340 DASD at address 341. You must respond to the
| message

| ENTER DEVICE ADDRESS WHERE SYSTEM RESIDENCE WILL BE BUILD (cuu):

| with the same device address you specified on the SYSRES operand cf the
| SYSRES macro in your CP system control file (DMKSYS).

| If you are doing the system generation in a virtual machine, you must
| respond "no" to the

| CHANGE TCD CLCCK (YES|NO):

| message.

| You must also respond to messages to set the date and time. If you
| are using a 3270 display device to generate VM/370 from the starter
| system for the first time, the 3270 locks when you try to respond to the
| message

| SET DATE mm/dd/yy:

| The System Available and Input Inhibited lights are on. You must press
| the Reset key and move the cursor to the second position of the input
| area before responding to the message. You follow this procedure only
| the first time the system is defined.

| When asked what kind of start this is, you must respond cold. The
| messages you receive at this time are:

```
|  VM/370 STARTER SYSTEM VERSION 2.0

|  ENTER PRINTER ADDRESS (CUU):00e
|  ENTER DEVICE TYPE (1403,1443,3211):1403
|  ENTER PUNCH ADDRESS (CUU):00d
|  ENTER DEVICE TYPE(2540P,3525):2540p
|  ENTER READER ADDRESS (CUU):00c
|  ENTER DEVICE TYPE (2501,2540R,3505):2540r
|  ENTER ADDRESS WHERE PID TAPE IS MOUNTED (CUU):280
|  ENTER DEVICE TYPE (2401,2415,2420,3420):2401
|  ENTER ADDRESS WHERE SCRATCH TAPE IS MOUNTED (CUU):281
|  ENTER DEVICE TYPE (2401,2415,2420,3420):2401
|  ENTER DEVICE ADDRESS WHERE SYSTEM RESIDENCE WILL BE BUILT (CUU):341
|  ENTER DEVICE TYPE (2319,2314,3340,3330,2305):3340

|  ***SYSTEM DEFINITION COMPLETED***

|  00E PRINTER
|  00D PUNCH
|  00C READER
|  280 PID TAPE
|  281 SCRATCH TAPE
|  341 NEW SYSTEM RESIDENCE
|  ARE THE ABOVE ENTRIES CORRECT (YES,NO):yes

|  VM/370 VERSION vv LEVEL 0 PLC 0000; mm/dd/yy hh:mm:ss
|  NOW 08:54:23 EDT FRIDAY mm/dd/yy
|  CHANGE TOD CLOCK (YES|NO):yes
|  SET DATE MM/DD/YY :mm/dd/yy
|  SET TIME HH:MM:SS :09:04:36
|  PRESS "TOD ENABLE SET" KEY AT DESIGNATED INSTANT

|  NOW 09:04:36 EDT FRIDAY mm/dd/yy
|  CHANGE TOD CLOCK (YES|NO):no
|  09:04:38 START ((COLD|WARM) (DRAIN))|(SHUTDOWN):cold
```

```
09:04:40
09:04:42 LOGON AT 09:04:37 EDT FRIDAY mm/dd/yy
LINE 01F LOGON AS CPGEN      USERS= 001
09:04:42

DMKCPI952I nnnnK SYSTEM STORAGE
```

If you have not defined your system residence volume with a label of VMREL2, three messages are issued indicating that the volume labeled VMREL2 is not mounted. If you have labeled it as VMREL2, one or more messages are issued indicating VMREL2 conflict. You can ignore the following messages.

```
09:04:43 FILES:  NO RDR,  NO PRT,  NO PUN
09:04:44

DMKIOG556I FORMATTING I/O ERROR RECORDING MASK

09:04:46

DMKIOG557I FORMATTING MCH ERROR RECORDING AREA
```

The TOD clock referred to is the System/370 Time of Day clock. Enter the actual date and time in response to the "SET DATE" and "SET TIME" messages, and press the TOD Enable Set switch on the system control panel when the exact time specified agrees with the installation wall clock.

If, for some reason, you must reload the starter system, the following message is displayed:

```
VM/370 STARTER SYSTEM VERSION 2.0

DO YOU WISH TO RE-DEFINE YOUR SYSTEM (YES|NO):
```

Respond by entering "no", unless the failure was the result of incorrectly specifying the device addresses, or unless the tapes and/or disks have been moved. If you reply "yes", you repeat Step 7.

STEP 8. SET THE TERMINAL MODE AND SPOOL THE CONSOLE

If the system console you are using is a display device, you should at this point, spool your console output so that you have a record of what you do. To spool the console input and output, issue the command:

```
spool console start
```

to save a copy of the system generation.

Because the default terminal environment for the primary system operator is CP, you should also issue the command:

```
terminal mode vm
```

The virtual machine terminal mode lets you remain in the CMS environment when you enter data on the display device.

STEP 9. DEFINE OR ATTACH THE SYSTEM RESIDENCE DEVICE

The CPGEN virtual machine assumes that the system residence volume is labeled VMREL2 and is at virtual address 350 if the device type is 3340, at virtual address 351 if the device type is 3330, or at virtual address 352 if the device type is 2314. If you labeled your system residence volume VMREL2 in Step 2, your system residence device is already available; now you, as the operator of CPGEN, must define it. Use Procedure 1 to define your system residence device.

If you did not label your system residence device VMREL2, you must attach it to your virtual machine, CPGEN, now. Use Procedure 2 to attach your system residence device.

For example, if you used the values shown in Step 7, you designated the 341 drive, labeled VMREL2, as your system residence drive for the generation procedure. Now you must define your virtual device 350 so that it corresponds to a real disk 341.

Use one of the following procedures to define or attach your system residence volume.

- Procedure 1

  If you labeled your system residence volume VMREL2, but did not define its address as 350, you must do so now. Press the Request key and enter the command:

  ```
  define 350 as 341
  ```

  You, as the operator of the CPGEN virtual machine, gain ownership of that entire volume as virtual address 350. The CP DEFINE command maps this virtual address to the real address of your residence system volume as defined in your DMKSYS module.

  Remember that you restored your starter system to a 3340 device. If you are now creating a 3330 system residence, with label VMREL2, issue the command:

  ```
  define 351 as 341
  ```

  or, if you are creating a 2314 system residence, with label VMREL2, issue the command:

  ```
  define 352 as 341
  ```

- Procedure 2

  If you did not label your system residence volume as VMREL2, you must attach your system residence volume to your virtual machine now. Press the Request key and enter the command:

  ```
  attach 341 to cpgen as 341
  ```

  Note: The first device address you specify is for the real device; the second device address is for the virtual device. The real 341 is the system residence volume you formatted in Step 2. The virtual 341 is the system residence device defined in DMKSYS (with the SYSRES macro) and also entered in response to the message:

  ```
  ENTER DEVICE ADDRESS WHERE SYSTEM RESIDENCE WILL BE BUILT (CUU):
  ```

STEP 10. MAKE OTHER DEVICES AVAILABLE

You must attach your tape drives and designate the printer to receive abnormal termination dumps if they occur. Press the Request key to enter each of the following commands:

```
attach 280 to cpgen as 181
attach 281 to cpgen as 182
set dump 00e
```

In the first command, the real tape drive (280) attached must be the same real address you entered in Step 7 in response to the message:

```
ENTER ADDRESS WHERE PID TAPE IS MOUNTED (CUU):
```

This real device must be attached to CPGEN as 181, because the VMFBLD EXEC procedure (which is used later) expects it to be 181.

In the second command, the real tape drive (281) attached must be the same real address you entered in Step 7 in response to the message:

```
ENTER ADDRESS WHERE SCRATCH TAPE IS MOUNTED (CUU):
```

This real device must be attached to CPGEN as 182, because the GENERATE EXEC procedure (which is used later) expects it to be 182.

If only one tape drive is available, enter

```
attach 280 to cpgen as 181
```

You can define your virtual 181 as 182 later in the system generation procedure.

The address of the real printer, defined in Step 7, is 00E. Any system dumps that occur are directed to that address.

If you wish, you can issue the CP command:

```
query virtual all
```

before and after performing Step 10, to see how your virtual machine's configuration changes.

STEP 11. LOAD CMS

Next, you must load CMS from virtual address 190 by issuing the CP command:

```
ipl 190
```

After CMS is loaded, a message is displayed on the system console indicating that CMS has successfully loaded:

```
CMS VERSION 2.0 - mm/dd/yy hh.mm
```

Press the Return key and the CPGEN 191 minidisk is accessed as your A-disk.

STEP 12. LOAD THE PLC (PROGRAM LEVEL CHANGE) TAPE

Demount the PID (Program Information Department) starter system tape, and mount and ready the PLC tape at 280.

The PLC tape is prepared by the VM/370 Program Level Change (PLC) service. This is a system update service that can include new functions as well as cumulative system changes. The latest PLC tape contains all new updates, as well as all previous updates since the last VM/370 release base. IBM Field Engineering is responsible for initially ordering the PLC service. Thereafter, PID automatically ships the PLC tapes to the FE location, and FE is responsible for applying the updates to VM/370 systems.

The PLC tape supplied with the starter system should be installed as part of the system generation procedure. Issue the CMS command:

```
tape load
```

The system issues the following messages:

```
LOADING.....

VMFBLD   EXEC     A1
END-OF-FILE OR END-OF-TAPE
R;
```

The VMFBLD EXEC procedure is a service program that adds or replaces CP, CMS, or RSCS object modules. It loads the updates from the PLC tape for you. Issue the command:

```
access 191 c
```

because VMFBLD accesses other disks as its A-disk during processing. Next, issue the command:

```
vmfbld cp
```

to execute VMFBLD.

The VMFBLD EXEC prompts you with the message:

```
ENTER CP STAGING AREA DISK ADDRESS:
194
```

You respond with 194. The 194 minidisk belongs to the CPGEN virtual machine and is the area on the restored starter system volume that contains all the object modules necessary to generate a CP nucleus. The contents of the minidisk are updated with the latest versions of all object modules, as well as the latest version of the DMKMAC macro library. The following message is displayed:

```
ARE YOU A NEW USER? -- RESPOND (YES|NO)
```

You must respond "yes" to this question. The following messages are then displayed:

```
'190' REPLACES 'A(194)'
190 ALSO = S-DISK
hh:mm:ss REWIND COMPLETE
CMS VERSION v.0 - mm/dd/yy hh.mm
```

The VMFBLD EXEC procedure links to the CMS system disk and accesses it as the A-disk. Next, VMFBLD reads the updated service programs from the PLC tape and writes them to the CMS S-disk. Then VMFBLD loads the new CMS system onto the system disk.

After CMS is loaded, a message is displayed to indicate that the CMS system was successfully loaded:

```
CMS VERSION v.0 - mm/dd/yy hh.mm
```

You must enter a null line to continue.

STEP 13. PUNCH THE SERVICE PROGRAMS

Use the GENERATE EXEC procedure to punch the service programs. Issue the command:

```
generate srvcpgm
```

These service programs are needed for standalone use; you should externally identify the decks and keep them intact.

The programs punched are:

- DMKFMT (A three-card loader precedes the DMKFMT text deck)
- DMKDIR (A three-card loader precedes the DMKDIR text deck)
- DMKDDR (A three-card loader precedes the DMKDDR text deck)
- IBCDASDI (The IBCDASDI text deck is loadable)

The GENERATE EXEC issues the following message:

```
THE FOLLOWING STANDALONE SERVICE PROGRAMS ARE BEING PUNCHED
      ** FORMAT - DIRECT - DUMP/RESTORE - IBCDASDI **

PUNCHING ' IPL FMT ' ******

PUNCHING ' IPL DIR ' ******

PUNCHING ' IPL DDR ' ******

PUNCHING ' IPL IBCDASDI ' ******
```

Each program deck is preceded by a CP userid card and several separator cards, all of which may be discarded. The format of these cards is described in the VM/370: Operator's Guide.

For more information about the GENERATE EXEC procedure, see "Part 6: Updating VM/370."

STEP 14. PRINT AND PUNCH THE STARTER SYSTEM SUPPLIED DIRECTORY, DMKSNT, DMKSYS, AND DMKFCB

After the service programs are punched, the GENERATE EXEC asks you whether you want a copy of the Release 2 directory printed. You should respond "yes."

```
PRINT COPY OF RELEASE2 DIRECT? -- RESPOND (YES|NO): yes
```

This prints a copy of the Release 2 directory, as well as copies of the DMKSYS ASSEMBLE (the CP system control file), DMKFCB ASSEMBLE (the forms control buffer file), and DMKSNT ASSEMBLE (the system name table) provided with the starter system.

The GENERATE EXEC procedure issues the following message:

```
A SAMPLE DIRECTORY IS BEING PRINTED TO AID YOU.
IT SHOWS WHERE THE VIRTUAL DISKS ARE LOCATED ON 'CPV2L0'
YOU MAY USE THESE MINIDISKS FOR OTHER VIRTUAL MACHINES,
IN PARTICULAR THE CMS SYSTEM DISK ( MAINT 190 ) AND
THE CP STAGING AREA DISK ( MAINT 194 )
INCLUDED IN THIS DIRECTORY IS THE USERID: MAINT
WHICH WILL BE USED FOR FUTURE SUPPORT OF THE SYSTEM.
THIS USERID SHOULD BE INCLUDED IN THE DIRECTORY
YOU BUILD FOR YOUR FLOOR USE.
** CAUTION ** IF YOU DESTROY USER MAINT'S AREAS, IT WILL BE
              NECESSARY TO RE-BUILD THE ENTIRE SYSTEM.

A SAMPLE OF DMKSYS, DMKFCB, AND DMKSNT ASSEMBLE ARE ALSO BEING
PRINTED TO AID YOU. THIS SAMPLE DMKSNT IS BASED ON THE
INFORMATION INCLUDED IN THE SAMPLE DMKSYS AS WELL AS THE
EXAMPLE ALLOCATIONS FOR VMREL2 PROVIDED IN THE SYSGEN GUIDE.
A COPY OF THIS DMKSNT MODULE HAS BEEN INCLUDED IN THE CP NUCLEUS,
SUCH THAT IF ONE USES THE INCLUDED DMKSYS AND THE
SAMPLE ALLOCATION PROVIDED IN THE SYSTEM GENERATION GUIDE,
HE WILL BE ABLE TO SAVE HIS CMS SYSTEM UPON COMPLETION
OF THE SYSTEM GENERATION PROCEDURE. A COPY OF DMKFCB HAS BEEN
INCLUDED IN THE NUCLEUS AND NEED NOT BE RE-ASSEMBLED FOR
SYSTEM GENERATION. IT HAS BEEN INCLUDED FOR THE USER WHO WOULD LIKE
TO MODIFY OR ADD TO THE EXISTING BUFFER LOAD.

NOTE: IF THE USER WISHES TO MODIFY THE SAMPLE DMKSNT AND/OR DMKFCB
HE MAY INCLUDE THE UPDATED SOURCE WITH THE SOURCE INCLUDED UNDER
THE OPTION 'GENERATE VM370', OF THE SYSTEM GENERATION PROCEDURE.
IF PRESENT, IT WILL AUTOMATICALLY BE ASSEMBLED AND INCLUDED IN THE
NEW CP NUCLEUS.
```

Again, the GENERATE EXEC procedure prompts you:

```
DO YOU WISH TO HAVE A COPY OF DMKSNT, DMKSYS, DMKFCB, AND
RELEASE2 DIRECT PUNCHED TO CARDS? -- RESPOND (YES|NO):
```

You should enter "yes" to have these decks punched. "Part 2: Defining Your VM/370 System" contains a listing of the Release 2 VM/370 directory and the DMKSNT, DMKSYS, and DMKFCB modules supplied with the starter system. The following messages are issued to indicate the files that are being punched:

```
PUNCHING ' DMKSNT ASSEMBLE ' ******

PUNCHING ' DMKSYS ASSEMBLE ' ******

PUNCHING ' DMKFCB ASSEMBLE ' ******

PUNCHING ' RELEASE2 DIRECT ' ******
R;
```

STEP 15. BUILD THE VM/370 DIRECTORY AND ASSEMBLE THE FILES DEFINING THE REAL I/O AND SYSTEM DEVICES

Next, you should place the files describing your installation's version of the VM/370 directory, real I/O configuration, and system devices in the reader and invoke the GENERATE EXEC procedure to build the directory and assemble the files describing the configuration. Place the following cards in the card reader, in the sequence shown:

```
ID CPGEN
:READ  filename DIRECT
(Directory program control statements)
:READ  DMKRIO ASSEMBLE
(Real I/O configuration macros)
:READ  DMKSYS ASSEMBLE
(CP system control file macros)
```

The Directory program control statements, real I/O configuration macros and system control macros are those you created according to the instructions in "Part 2: Defining Your VM/370 System." You can code your own system control macros or add to the file supplied with the starter system.

Also, if you want, you can change the printer buffer load (DMKFCB) or the named system (DMKSNT)modules to conform to local requirements. If you want to change the printer buffer load or the system name table, you should add your changes to the files already punched (in Step 14) and place the following files in the card reader behind the system control file macros:

```
:READ DMKFCB ASSEMBLE
(forms control macros)
:READ DMKSNT ASSEMBLE
(system name table macros)
```

Instructions for creating new forms control macros are in the VM/370: System Programmer's Guide. The system name table macros are described in "Part 2: Defining Your VM/370 System" of this manual.

FORMAT OF THE READ CONTROL STATEMENT

The READ control statements must be punched according to the following format:

Column	Number of Characters	Contents	Meaning
1	1	colon':'	Identifies card as a control card
2-5	4	READ	Identifies card as a READ control card
6-7	2	blank	
8-15	8	fname	Filename of the file punched
16	1	blank	
17-24	8	ftype	Filetype of the file punched
25-80	56	blank	

SPECIAL PROCEDURE IF YOU ARE USING ONLY ONE TAPE DRIVE

If you are using only one tape drive, at this time you must issue the command:

```
define 181 as 182
```

Now mount the scratch tape in place of the PLC tape and ready the device.

INVOKE THE GENERATE EXEC PROCEDURE

When all the files are placed in the reader, invoke the GENERATE EXEC procedure by issuing the following command:

```
generate vm370
```

This procedure invokes the Directory service program to build the disk-resident VM/370 directory, then assembles the DMKRIO and DMKSYS files that you placed in the real card reader. After the Directory service program execution completes, the DMKRIO and DMKSYS files are assembled in preparation for building the new CP system nucleus. GENERATE then checks for DMKFCB and DMKSNT source files. When new versions of the DMKFCB and DMKSNT modules are provided, GENERATE assembles the new modules and replaces the corresponding modules supplied with the starter system with the new modules. If any errors occur while the VM/370 directory is being built, the Directory program issues error messages and the GENERATE EXEC procedure issues the following message:

```
CORRECT THE DIRECTORY CARDS AND RELOAD THE CARD READER
  RESPOND WITH: GENERATE DIRECT
```

Correct the errors in the Directory program control statements, and reload the card reader with only the ID card, the :READ statement, and the Directory program control statements. Then respond with:

```
generate direct
```

If errors are detected while the DMKRIO, DMKSYS, DMKFCB, or DMKSNT files are assembling, GENERATE issues a similar message:

```
CORRECT THE filename ASSEMBLE FILE AND RELOAD THE CARD READER
  RESPOND WITH: GENERATE filename
```

You must correct the errors in the indicated file, and reload the card reader with only the ID card, the :READ statement, and the appropriate file. Then, issue the GENERATE command with the appropriate option. For example:

```
generate dmksys
```

STEP 16. SPECIAL CONSIDERATIONS FOR VIRTUAL=REAL MACHINES

Once the directory is built and the files are assembled, the GENERATE EXEC procedure asks if you want the virtual=real option included in your CP nucleus:

```
VIRTUAL=REAL OPTION REQUIRED (YES,NO):
```

If you respond "yes" to this question, you are prompted to enter the amount of storage you wish to reserve in the CP nucleus for a virtual=real machine. ("Part 1. Planning for System Generation" contains formulas to help you determine how much storage you need to reserve.) This size must be a multiple of 4K and must be entered in the

format nnnnK. The minimum size you can specify is 32K and the maximum is 4M. For example:

```
STORAGE SIZE OF VIRT=REAL (MINIMUM IS 32K):
100k
0100K STORAGE SIZE FOR VIRTUAL=REAL
IS THE ABOVE ENTRY CORRECT (YES,NO):
yes
```

You must have enough virtual storage available to generate the CP nucleus with a virtual=real area. The "Specifying a Virtual=Real Machine" section of Part 1 tells you how to determine the amount of virtual storage you require. If you do not have enough virtual storage available, redefine storage for your virtual machine.

STEP 17. LOAD THE CP NUCLEUS

Once you respond to the virtual=real generation questions, the GENERATE EXEC procedure writes the CP nucleus on tape. To do this, GENERATE first issues the VMFLOAD command to punch the loader and CP object modules to a virtual punch spool file. Then it transfers this file to the virtual card reader file and writes the file on tape. GENERATE issues the following messages during the processing:

```
hh:mm:ss NO FILES PURGED
SYSTEM LOAD DECK COMPLETE
hh:mm:ss PUN FILE nnnn TO CPGEN COPY 01 NOHOLD
IPLABLE NUCLEUS NOW ON TAPE ****
hh:mm:ss PRT cuu OUTPUT OF CPGEN FILE=nnnn RECDS=nnnnnn
  COPY = 01 A PRT
```

Note: If any errors are detected while the tape is being written, you must recreate the CP nucleus. To do this, issue the command:

```
generate cp nucleus
```

The procedure restarts at Step 16 where you are asked if you want the virtual=real option.

LOADING A CP NUCLEUS WITHOUT A VIRTUAL=REAL AREA

If you responded "no" when asked if you wanted the virtual=real area, the GENERATE EXEC procedure loads the nucleus for you. You receive the following message:

```
WHEN 'NUCLEUS LOADED ON XXXXXX' IS TYPED, ISSUE 'CLOSE PRT',
  TO GET THE CPLOAD MAP.  WHEN PRINTING IS COMPLETE,
  SHUTDOWN THE SYSTEM AND IPL THE NEW SYSRES VOLUME.
```

When the nucleus is written on the system residence volume, the message

```
NUCLEUS LOADED ON volid
```

is issued, where volid is the volume serial number of your system residence volume. The volid is the serial number you specified on the SYSRES macro when you prepared the CP system control file (DMKSYS). If you followed the example in this manual, the serial number of your system residence volume is VMREL2.

To print the load map you must close the printer; issue the command:

```
close prt
```

When printing is complete, shutdown the system in preparation for loading your newly generated CP nucleus:

```
shutdown
```

LOADING A CP NUCLEUS THAT HAS A VIRTUAL=REAL AREA

If you responded "yes" when asked if you wanted the virtual=real area, the GENERATE EXEC procedure issues the following message:

```
TO LOAD THE CP NUCLEUS JUST CREATED, SHUTDOWN THE SYSTEM AND
THEN IPL THE TAPE. THE CPLOAD MAP WILL AUTOMATICALLY BE PRINTED AT THE
PRINTER WHOSE ADDRESS IS '00E'. IF THERE IS NO PRINTER AT THIS ADDRESS
THE LOAD MAP WILL BE PRINTED AT THE FIRST PRINTER CAUSING AN INTERRUPT
(IE. NOT-READY TO READY SEQUENCE). ONCE THE NUCLEUS HAS BEEN LOADED,
YOU MAY IPL YOUR NEW CP SYSTEM RESIDENCE VOLUME.

NOTE: THERE MUST BE ENOUGH STORAGE ON THE SYSTEM (VIRTUAL OR REAL) TO
CONTAIN THE VIRT=REAL AREA AND THE CP NUCLEUS.
```

You must be sure that there is enough storage to load the CP nucleus with a virtual=real area before you load it. The "Specifying a Virtual=Real Machine" section of Part 1 tells you how to determine the amount of real or virtual storage you need. You can load a CP nucleus that has a virtual=real area in either a real or virtual machine.

You IPL the tape containing the CP nucleus next. The CP nucleus is on the tape at virtual address 182 for the CPGEN virtual machine.

STEP 18. IPL THE NEWLY GENERATED VM/370

IPL the newly created system residence volume by setting the load unit address dials to the address of the system residence volume and pressing the Load button. The userid OPERATOR (or whatever userid you specified as the system operator on the SYSOPER macro) is logged on when you IPL. The following messages are issued. Respond as shown.

```
VM/370 VERSION vv LEVEL 00 PLC nnnn; mm/dd/yy hh:mm:ss

NOW hh:mm:ss EDT day mm/dd/yy
CHANGE TOD CLOCK (YES|NO):no
hh:mm:ss START ((COLD|WARM) (DRAIN))|(SHUTDOWN): cold
hh:mm:ss LOGON AT hh:mm:ss EDT day mm/dd/yy
hh:mm:ss LINE 01F LOGON  AS OPERATOR USERS = 001
hh:mm:ss

DMKCPI952I nnnnK SYSTEM STORAGE

hh:mm:ss FILES:  NO RDR,  NO PRT,  NO PUN
hh:mm:ss

DMKIOG556I FORMATTING I/O ERROR RECORDING AREA
```

```
hh:mm:ss

DMKIOG557I FORMATTING MCH ERROR RECORDING AREA
```

At the time you IPL your newly generated VM/370, your system residence volume is formatted according to your specification on the SYSRES macro. Also, if you used the VM/370 directory supplied with the 3340 starter system, your starter system volume (CPV2L0) is set up as shown in Figure 22. Be careful not to modify the 190 and 194 minidisks for the MAINT user. These minidisks and the information they contain are required for applying PLC (Program Level Change) updates to VM/370.

Real Cylinder	Number of Cylinders	Contents
0	1	Unused
1-2	2	191 minidisk for the IVPM1 user
3-4	2	191 minidisk for the IVPM2 user
5-6	2	191 minidisk for the REM2780 user
7-20	14	191 minidisk for the OPERATOR user
21-26	6	191 minidisk for the CE user
27-40	14	191 minidisk for the MAINT user
41-45	5	191 minidisk for the ECMODE user
46-47	2	199 minidisk for the MAINT user
48-250	203	190 minidisk for the MAINT user Note: The last two cylinders of this minidisk contain the CMS nucleus.
251-348	98	194 minidisk for the MAINT user

Note: Cylinders 349-697 are not used for a 3340 Model 70 system residence volume.

Figure 22. Allocation of the Starter System Volume (CPV2L0) When the 3340 Starter System Directory Is Used

STEP 19. BACK UP THE NEWLY GENERATED VM/370

At this time, you should be sure to back up your new system residence volume. If your real machine has at least 448K bytes of real storage, the tape created in Step 17 is sufficient backup. However, that tape is not sufficent backup for real systems with less than 448K of real storage because that tape cannot be loaded on such systems.

VM/370 systems that run on a real machine with less than 448K bytes of real storage should use the DASD Dump Restore (DDR) service program to create a backup tape similar to the one created in Step 17. The DASD dump restore program is described in Appendix F. If your system residence volume is at address 341 and you labeled it VMREL2, you could use the following DDR control statements to back it up:

```
input 341 3340 vmrel2
output 180 3420
dump nucleus
```

The tape produced contains the CP nucleus; it does not contain the VM/370 directory. However, you already created a standalone version of the Directory program in Step 13 and have the Directory program control statements, which you used in Step 15, to recreate the VM/370 directory.

If you do not wish to use the DDR program to back up your system, you can load the tape produced in Step 17 in a virtual machine. If you load the tape in a virtual machine that virtual machine must have (1) 512k of storage and (2) write-access to the system residence volume, at the address defined for system residence in the SYSRES macro of the CP System Control (DMKSYS) file.

When you use the DDR program to back up your system you do not get a load map when you restore the tape. You do get a load map if you load the tape produced in Step 17.

STEP 20. MOVE THE CMS SYSTEM DISK, IF YOU WISH

The CP nucleus just created is tailored to your installation's exact I/O configuration with all the appropriate options selected. The CMS nucleus requires no special tailoring for installation. The CMS system residence disk was created when the VM/370 starter system was loaded; it contains a copy of the CMS nucleus and disk-resident command modules. At this point, the CMS system resides on the virtual 190 disk for the user, MAINT. The operator's VM/370 directory has an entry that links to MAINT's 190 disk, therefore you can invoke CMS by issuing the command:

```
ipl 190
```

and thus have access to most of the CMS functions. Although most of the CMS functions are available, CMS has not yet been updated with the latest PLC changes. The Assembler that is available now is the one supplied with the starter system.

If you wish, you can move the CMS system disk to a disk other than the one designated by the starter system. To move the CMS system disk to another minidisk, use the COPY function of the DDR service program. The new minidisk should be the same size as the CMS system disk, but it does not need to be formatted. Also, if you move the CMS system, you must make the appropriate changes to the userid MAINT in your VM/370 directory (that is, change the volume label on the MDISK entry for 190).

If you add IBM Program Products or your own disk-resident modules to the CMS system disk, you may have to create a new system disk to contain the additional modules. The "Creating a CMS System Disk" section of "Part 6. Updating VM/370" describes this procedure.

STEP 21. FORMAT THE OPERATOR'S VIRTUAL 191 DISK

Before any new minidisk area can be used for CMS files, it must be initialized with the CMS FORMAT command, which formats the area into 800-byte blocks. Take care not to format areas which contain data restored from the starter system (such as the 190 and 194 minidisks

belonging to the user MAINT.) The CMS FORMAT command is described in the VM/370: Command Language Guide for General Users.

Note: After you complete this step, a portion of the starter system is overlaid by the operator's virtual 191 minidisk. If for any reason you wish to IPL the starter system again, you must restart at step 1.

At this time you are logged on as the operator. Use the following procedure to format your virtual disk 191. First, load CMS (IPL 190). CMS responds with:

```
CMS VERSION 2.0 - mm/dd/yy hh:mm
```

Next, enter the following command:

```
access (nodisk
```

The NODISK option prevents CMS from automatically accessing your virtual disk 191. (Accessing 191 at this time would causes an error message to be issued because 191 is not yet initialized, and therefore cannot be used.) After the Ready message is displayed, issue the command:

```
format 191 a
```

The CMS FORMAT command prompts you with the following message:

```
FORMAT WILL ERASE ALL FILES ON DISK 'A(191)'. DO YOU WISH TO
  CONTINUE? (YES|NO):
```

If you respond yes, CMS prompts you with:

```
ENTER DISK LABEL:
opr191
```

Enter the one- to six-character alphameric label of the virtual disk. You can use whatever label you wish for this virtual disk. In this example, the label is OPR191. CMS then issues:

```
FORMATTING DISK 'A'.
'14' CYLINDERS FORMATTED ON 'A(191)'.
```

and a Ready message.

STEP 22. FORMAT THE MAINT USER'S 191 DISK

Next, you should initialize the 191 disk belonging to MAINT, in the same way you initialized the operator's 191. Log yourself off the system and log on again as userid MAINT, using the password CPCMS. Now IPL 190. When the CMS Ready message is displayed, issue

```
access (nodisk
```

and continue to format MAINT's 191, using the same procedure you used to format OPR191.

STEP 23. UPDATE CMS

Now that the MAINT 191 minidisk is formatted, you (as the MAINT user) can update CMS with the changes supplied on the PLC tape. Mount the PLC tape, if it is not already mounted. Then, you must attach the real tape drive to your CMS virtual machine (MAINT); for example:

```
attach 280 to maint as 181
```

and, with the 191 minidisk that belongs to MAINT accessed as your A-disk, rewind and load the tape:

```
tape rew
tape load
```

CMS responds with the following message:

```
LOADING.....
VMFBLD   EXEC   A
END-OF-FILE OR END-OF-TAPE
```

BUILDING A NEW CMS

Now execute the VMFBLD EXEC procedure by issuing the commands:

```
access 191 c
vmfbld cms
```

The VMFBLD EXEC procedure prompts you with:

```
ENTER CMS SYSTEM DISK ADDRESS:
```

You respond with "190". Next, VMFBLD issues the message:

```
190 ALSC = S-DISK.
```

You are asked if you want the service programs punched:

```
PUNCH STANDALONE SERVICE PROGRAMS--RESPOND (YES|NO):
```

The following messages are displayed; respond as shown:

```
hh:mm:ss  NC  FILES PURGED
hh:mm:ss  NO  FILES PURGED
SYSTEM LCAD DECK COMPLETE
hh:mm:ss PUN FILE nnnn  TO  CPGEN    COPY 01 NOHOLD
hh:mm:ss REWIND COMPLETE
          WHEN THE NEW CMS SYSTEM IS BUILT ISSUE THE FOLLOWING
          COMMAND
               CLOSE PRT . . . (PRINTS THE LCAD MAP)

DMSINI606R SYSTEM DISK ADDRESS = 190
DMSINI615R Y-DISK ADDRESS = 19e
DMSINI607R REWRITE THE NUCLEUS ? yes
DMSINI608R IPL DEVICE ADDRESS = 190
DMSINI609R NUCLEUS CYL ADDRESS = 201
DMSINI610R ALSO IPL CYLINDER 0 ? yes
DMSINI611R VERSION IDENTIFICATION =
DMSINI612R INSTALLATION HEADING =
CMS VERSICN 2.0 - mm/dd/yy  hh:mm
```

In the example, the 190 you entered is the 190 minidisk that belongs to the user MAINT; it is equivalent to the 190 minidisk that belongs to CPGEN. If you used the VM/370 directory supplied with the starter system, you allocated 203 cylinders (virtual cylinders 0-202) to the 190 minidisk belonging to the user MAINT. The last two cylinders of this minidisk contain the CMS nucleus, therefore you specify cylinder 201 as the nucleus cylinder address when responding to message DMSINI609R.

When the VMFBLD EXEC procedure completes its execution, the CMS system residence volume is updated with the most current object modules (text decks) and load modules, and the new CMS nucleus is written on the CMS system residence volume.

Also, if there are any updates for the EREP program or the Assembler on the PLC tape, the VMFBLD EXEC procedure updates those programs and creates the corresponding auxiliary directories.

STEP 24. SAVE CMS

If you used the sample DMKSNT and DMKSYS supplied with the starter system, and used the sample allocations shown in Step 2, you can now save your CMS system.

To save the CMS system, you must load it and then save it as soon after loading as possible. Issue the command:

```
ipl 190
```

When the terminal unlocks, do not press carriage return, but immediately issue the command:

```
savesys cms
```

Then press carriage return. If CMS is successfully saved, the message

```
hh:mm:ss SYSTEM SAVED
CMS VERSION 2.0 - mm/dd/yy hh:mm
```

is displayed. Your CMS system is now saved; you can issue IPL CMS instead of IPL 190, when you wish to run CMS.

If you named your CMS system something other than CMS, such as CMS1, the entry you made in the system name table would be for CMS1, you would have to save CMS1 (SAVESYS CMS1) and then you could IPL CMS1. For more information about how to save CMS, see the VM/370: System Programmer's Guide or the "Saved Systems" section of this manual.

At this point, use the Installation Verification Procedure (IVP) to test the new system. Log off as the userid MAINT and log on again as the userid OPERATOR, using the password OPERATOR. The IVP description follows.

Verifying CP and CMS Using the IVP

The Installation Verification Procedure (IVP) for VM/370 exercises CP and CMS to verify that they are working properly. The IVP is contained in two files using the EXEC facility of CMS, and uses two virtual machines in addition to the system operator's virtual machine.

The tests exercise the following areas of CP:

- Multiple virtual machine support
- I/O spooling
- Transferring of spooled data to other virtual machines
- Offline I/O operations
- Sending of messages to the system operator
- Paging operations
- Task dispatching and scheduling
- Disk I/O support
- Automatic warm start following abnormal termination of VM/370

The following facilities of CMS are exercised:

- Normal CMS command processing
- Disk formatting
- Copying of files
- Creation and modification of files via EDIT command
- Assembly of executable programs
- Execution of user programs
- Creation and execution of user-written commands
- Printing and punching of CMS files
- Issuing of commands to CP
- Use of multilevel nested EXEC procedures
- Stacking and unstacking of command and data input from the terminal
- Communication with user from EXEC procedures

Several other system facilities, incidental to the primary IVP tests, are exercised. Certain system facilities such as preferred execution options, virtual=real, OS ISAM, remote 2780, and RSCS (Remote Spooling Communications Subsystem) are not exercised by the IVP. The IVP requires operator intervention only when an operational decision is to be made, or to initiate the IVP tests themselves. All file creation, erasure, management, and logoff of the virtual machines (with the exception of the system operator) at test completion is performed without operator or user action.

The IVP tests use only the system-provided facilities. All unique test programs are created, assembled, and subsequently erased by the IVP.

FACILITIES REQUIRED FOR EACH IVP VIRTUAL MACHINE

All VM/370 configurations are supported. The IVP executes under the control of CMS. The other facilities required are:

- The Assembler
- One virtual read/write disk accessed as the A-disk (normally at virtual address 191)
- 320K of virtual storage (16M for IVPM1)

STARTING THE IVP

The IVP must be executed to formally complete the initial installation. (See "Variations of the IVP" for post-installation testing.) It requires two virtual machines (IVPM1, IVPM2) which must be described in the VM/370 directory.

The directory entries for the IVP virtual machines, IVP1 and IVP2, are included in the VM/370 directory supplied with the starter system; these entries should be included in your own directory.

You, as the system operator, execute the IVP. To initiate the IVP tests, enter the command:

```
IVP
```

and then answer "yes" or "no" to the following question:

```
ARE YOU THE SYSTEM OPERATOR? ENTER "YES" OR "NO":
```

If you enter the IVP command with no parameters specified and then reply that you are not the system operator, the IVP tests default to the single virtual machine verification procedure. (See "Variations of the IVP.")

Prompting instructions are displayed whenever you must perform an operation or issue a command.

Log on the virtual machine (IVPM1), using the password IVPASS, and IPL CMS to continue the testing procedure:

```
logon ivpm1
ENTER PASSWORD:
ivpass
LOGON AT 09:55:00 EST FRIDAY 03/29/74
define storage 16384k
STORAGE=16384K
ipl 190
CMS VERSION 2.0 mm/dd/yy hh.mm
ivp 1
```

At this point, the tests begin on virtual machine 1. After the disconnect message is displayed, follow the prompting messages that are displayed. These messages tell you to log on the IVPM2 virtual machine (this may be done on the same terminal), as shown:

```
logon ivpm2
ENTER PASSWORD:
ivpass
LOGON AT 09:58:30 EST FRIDAY 03/29/74
ipl 190
CMS VERSION 2.0 mm/dd/yy hh.mm
ivp 2
```

At this point, the remainder of the tests begin on virtual machine 2. The final phase of the IVP tests consists of displaying, printing, and punching a file which contains the messages generated by IVPM1 following the DISCONNECT.

Upon completion of the tests, the IVP EXEC procedure logs off. The system abnormal termination test, which consists of forcing an ABEND dump of VM/370 and the subsequent warm start, is an option that you must specify in response to messages that are displayed. For the purpose of installation verification, you should select this option. You are instructed to delay starting the spooling devices (reader, printer, and punch) until after the warm start procedure.

VARIATIONS OF THE IVP

If you wish, you may run the IVP procedure after initial installation, using any one of the following methods:

- Executing the IVP without testing the system abnormal termination
- Executing the IVP using virtual machines other than IVPM1 and IVPM2
- Executing the IVP in a single virtual machine

When you execute the IVP in a single virutal machine, inter-machine functions, such as transferring data between virtual machines, are nct exercised.

To execute the IVP without testing system abnormal termination:

- Retain the created virtual machines in your VM/370 directory.
- Execute the IVP as described previously under "Starting the IVP," but do not select the "system abnormal termination" option.

To execute the IVP with virtual machines other than IVP1 and IVP2:

- Enter, in place of the command IVP 1:

```
IVP 1 userid1
```

- Enter, in place of the command IVP 2:

```
IVP 2 userid2
```

where userid1 and userid2 identify the two virtual machines in which the EXEC procedures IVP 1 and IVP 2 (respectively) are to be executed.

To execute the IVP in a single virtual machine, enter the command (from any logged-on virtual machine):

```
IVP *
```

This causes the IVP tests to be run in that single virtual machine. Inter-machine transfer of data is simulated by transferring virtual punched output to the same virtual machine's virtual card reader.

INTERPRETING THE TEST RESULTS

Messages at the end of the IVP test indicate successful completion. If any errors are detected by the IVP, notify the IBM FE Program Systems Representative, since an error usually indicates a serious malfunction of the generated system. The IVP procedure identifies each command being tested just before the command is executed.

Error messages are displayed in a four-line format, for example:

```
*** IVP FAILURE HAS OCCURRED ***
*** COMMAND: STATE IVPTST *
*** EXPECTED RETURN CODE 28
*** RECEIVED RETURN CODE 0
```

These messages indicate that the CMS STATE command had a return code of 0, instead of the expected 28.

All information messages that originate within the IVP are preceded by three asterisks (***).

If any command fails, the IVP procedure terminates. Follow the instructions (if any are given) to log off the virtual machine.

Generating and Installing RSCS

GENERAL INFORMATION

The data and control files required to generate and install RSCS are contained on the Program Level Change (PLC) tape of the VM/370 system. You can use the PLC installation EXEC procedure (VMFPLD) to load the RSCS data onto the CP staging area. Four modules must be moved from the CP staging area disk to the RSCS system disk: DMTAXS, DMTLAX, DMTNPT and DMTSML.

RSCS data, on the PLC tape, consists of the following:

File	Contents
DMTxxx TEXT	The pre-assembled nucleus modules and supervisor routines required to generate RSCS are: DMTAKE, DMTASK, DMTASY, DMTCMX, DMTCOM, DMTCRE, DMTDSP, DMTEXT, DMTGIV, DMTINI, DMTIOM, DMTMAP, DMTMGX, DMTMSG, DMTPST, DMTQRQ, DMTREX, DMTSIG, DMTSTO, DMTSVC, DMTVEC, and DMTWAT.
DMTAXS TEXT	The spool file access method supervisor task.
DMTLAX TEXT	The communication line allocation supervisor task.
DMTNPT TEXT	The Nonprogrammable Terminal (NPT) line driver module.
DMTSML TEXT	The Spool MULTI-LEAVING (SML) line driver module.
DMTLOAD EXEC	The loadlist EXEC file. This file is required to generate an RSCS nucleus on the RSCS system task.
DMTSYS ASSEMBLE	The RSCS configuration table module. This file must be assembled with the COPY files you create to define your RSCS configuration (the AXSLINKS, LAXLINES, and TAGQUEUE COPY files).
DMTMAC MACLIB	The file containing all the macros needed to assemble the RSCS source files.
DMTR20 CNTRL	The control file that is needed to assemble the configuration table via the VMFASM EXEC procedure.
DMTxxx ASSEMBLE	All the source files for RSCS. There is an ASSEMBLE file for each TEXT file previously listed.
DMTMAC EXEC	An EXEC file used to generate the DMTMAC MACLIB.

Also, the PLC tape contains all the macro and copy files included in the DMTMAC macro library.

DEFINING YOUR RSCS VIRTUAL MACHINE

You need the RSCS virtual machine defined when you generate RSCS. This virtual machine must have 512K of virtual storage, a console, and a 5-cylinder RSCS system disk with a write password.

If you did not include an entry in your VM/370 directory for the RSCS virtual machine, you must add one now. See "Part 2: Defining Your VM/370 System" for a description of the Directory program, including the control statements necessary to define a virtual machine. Add the newly-coded control statements to the existing directory file and assemble and load the new VM/370 directory. Use the GENERATE EXEC procedure to assemble and load your updated VM/370. GENERATE is described in "Part 6: Updating VM/370."

A suggested VM/370 directory entry for an RSCS virtual machine is:

```
USER RSCS password 512K
ACCOUNT NUMBER BIN17
CONSOLE 009 3215
SPOOL 001 2540 READER A
SPOOL C 2540 READER A
SPOOL D 2540 PUNCH A
SPOOL E 1403 A
LINK CMSSYS 190 190 R
MDISK 191 3330 81 5 UDISK1 W password
DEDICATE 0B1 078
DEDICATE 0B2 079
DEDICATE 0B3 07A
```

If you plan to run multiple concurrent RSCS virtual machines, you must define multiple RSCS virtual machines in your VM/370 directory. Also each RSCS virtual machine must have a unique system disk with a unique local location identification generated.

DEFINING YOUR RSCS SYSTEM

You must create three files to define your RSCS system:

File	What It Defines
AXSLINK COPY	All the possible paths for data transmission between your RSCS virtual machine and all the single uniquely identified remote stations.
LAXLINES COPY	The virtual device addresses of all the switchable telecommunications line ports dedicated to the RSCS virtual machine.
TAGQUEUE COPY	Total number of virtual storage tag slots. A tag slot contains spool file information needed by RSCS when it processes files.

These three files are used to generate the DMTLOC macro library during the RSCS installation procedure. The DMTLOC macro library is used when the RSCS configuration table module (DMTSYS) is assembled.

CREATING THE AXSLINKS COPY FILE

The AXSLINKS COPY file defines the set of links to be loaded in the RSCS configuration table module. A link is a potential path for data transmission between an RSCS virtual machine and a single uniquely identified remote station.

The AXSLINKS COPY file is a list of 1 to 64 GENLINK macro statements. The GENLINK macro defines the attributes of a link. The

first GENLINK macro in the AXSLINKS file must contain the ID of the local RSCS station. You must also code the TYPE=driverid operand with a valid filename on this first GENLINK macro. You should code additional GENLINK macros, with no operands, for links you may want to define temporarily during an operating session.

The format of the GENLINK macro is:

```
┌──────────────────────────────────────────────────────────────────────┐
│          │ ┌                                                        │
│ GENLINK  │ │ ID=linkid,TYPE=driverid[,CLASS=c]                      │
│          │ │                        [,KEEP=holdslot]                │
│          │ │                        [,LINE=vaddr]                   │
│          │ │                        [,TASK=taskname]                │
│          │ └                                                        │
└──────────────────────────────────────────────────────────────────────┘
```

where:

ID=linkid is a one- to eight-character alphameric location ID of the remote location to be served by the link. If this operand is not specified, the ID defaults to "undefined."

TYPE=driverid
is a CMS filename of a file which is the TEXT file for the line driver program to be used to process files for the link. The appropriate line driver program to be specified depends on the type of remote telecommunications facilities to be used.

The TYPE operand must be specified if ID=linkid is coded. If the TYPE operand is omitted, the TYPE defaults to "undefined".

CLASS=c is the spooling class(es) of the files which can be processed by the active link. You can specify up to four spooling classes (single alphameric characters from A to Z and 0 to 9) with no intervening blanks, or *, which means all spool file classes may be processed. If the CLASS operand is not specified, the default is "*".

KEEP=holdslot
is a decimal number from zero to 16 which designates the number of virtual storage file tag slots to be reserved for exclusive use by the link. If the KEEP operand is omitted, a default "holdslot" value of two is assumed.

LINE=vaddr
designates the virtual device address of a permanent telecommunications line port to be used for processing files on the link. If the LINE operand is omitted, the default is "undefined".

TASK=taskname
is a one- to four-character alphameric identifier. It specifies the task name to be used by the line driver associated with the link. If the TASK operand is omitted, the default is "undefined".

CREATING THE LAXLINES COPY FILE

The LAXLINES COPY file defines the virtual device addresses of the telecommunications line ports (attached to the RSCS virtual machine) which may be shared among the various active links. Such line ports must be switchable, and the links which are to use these line ports must have switched line ports available at their remcte stations.

The LAXLINES COPY file consists of a list of GENLINE macro statements, one for each line port. The format of the GENLINE macro is as follows:

```
GENLINE | LINE=vaddr
```

where:

LINE=vaddr
: is the virtual device address of a switched telecommunications line port available to RSCS.

CREATING THE TAGQUEUE COPY FILE

The TAGQUEUE COPY file specifies the total number of virtual storage tag slots to be available to RSCS. The format of the GENTAGQ macro is:

```
GENTAGQ | NUM=totslots
```

where:

NUM=totslots
: is a decimal number from 32 to 512 that defines the total number of virtual storage tag slots to be made available to RSCS for storing information on files enqueued for transmission or received from remote stations. Files which cannot be enqueued for transmission because no free virtual storage tag slots are available are left pending, and are automatically accepted and enqueued at a later time as virtual storage tag slots become available.

  You must specify a number for totslots that is at least as large as the sum of linkid tag slots defined or implied by the KEEP=holdslot operand of all the GENLINK macros.

GENERATION PRCCEDURE FOR RSCS

Before you perform the generation procedure, be sure you have the following:

- The VM/370 PLC tape
- A VM/370 directory entry for your RSCS virtual machine

- A VM/370 directory entry for the software system support virtual machine (for example, the MAINT entry supplied with the VM/370 starter system directory).

Also, be sure that the PLC tape containing RSCS was applied to your VM/370 system.

You can use a 2314, 3330, or 3340 disk as the RSCS system disk. The RSCS nucleus occupies two cylinders on a 2314 or 3340, and one cylinder on a 3330 disk.

The following system generation procedure for RSCS assumes you have the MAINT virtual machine supplied with the VM/370 starter system in your VM/370 directory.

STEP 1. LOG ON AS MAINT AND IPL CMS

To build the RSCS nucleus, you must log on the software system support virtual machine (MAINT) and IPL the CMS system.

STEP 2. ATTACH AND LOAD THE PLC TAPE

Mount the PIC tape, if it is not already mounted. Then you must attach the real tape drive to your CMS virtual machine (MAINT). For example:

```
attach 280 to maint as 181
```

The PLC tape must be at address 181 because the VMFBLD EXEC procedure expects to find it at that address.

With the 191 minidisk that belongs to MAINT accessed as your A-disk, rewind and load the tape:

```
cp rewind 181
tape load
```

CMS responds with the following message:

```
LOADING...
VMFBLD   EXEC     A
END-OF-FILE OR END-OF-TAPE
```

STEP 3. INVOKE VMFBLD TO LOAD THE RSCS FILES

Access the 191 minidisk belonging to MAINT as the C-disk and invoke VMFBLD:

```
access 191 c
vmfbld rscs
```

The VMFBLD EXEC procedure prompts you with the message:

```
ENTER CP STAGING AREA DISK ADDRESS:
194
```

You respond with 194. The 194 minidisk is the area where the RSCS object files are to be loaded. VMFBLD next loads all the RSCS text files and the DMTSYS ASSEMBLE file, as well as all the control files and macro libraries needed to generate the RSCS nucleus, onto the 194 minidisk.

When all the RSCS files are loaded, VMFBLD displays the message:

```
DO YOU WISH TO BUILD RSCS SYSTEM -- RESPOND (YES|NO)
```

You must respond "no" at this time.

STEP 4. FORMAT THE RSCS SYSTEM DISK

You must link to the RSCS system disk and format it. If you used the suggested VM/370 directory entry for the RSCS virtual machine, your LINK command is:

```
link to rscs 191 as 195 w pass= password
```

This makes the RSCS system disk (address 191 in the RSCS virtual machine) available at virtual address 195 in the MAINT virtual machine. Remember the address you specify for MAINT. You must use this same virtual address later in the RSCS generation procedure (you use 195 when you invoke VMFBLD again to build the RSCS nucleus in Step 8). Then, format the RSCS system disk.

```
format 195 a
```

Next, format the RSCS system disk again. You must use the recompute function of the FORMAT command to make the last cylinders of the RSCS system disk unavailable to the CMS file system. The last one or two cylinders contain the RSCS nucleus. If the RSCS system disk is a 2314 or 3340, the last two cylinders are needed for the nucleus. For a 3330, only the last cylinder is needed. The FORMAT command for a 2314 or 3340 RSCS system disk is:

```
format 195 a 3 (recomp
```

The FORMAT command for a 3330 RSCS system disk is:

```
format 195 a 4 (recomp
```

STEP 5. CREATE THE COPY FILES THAT DEFINE YOUR RSCS SYSTEM

You can use the CMS Editor to create the three COPY files you need: AXSLINKS, LAXLINES, and TAGQUEUE. The macros you need to include in these files are described in a preceding section, "Defining Your RSCS System."

When you code the GENLINK macros for the AXSLINKS COPY file, you may include GENLINK macros with no operands. These macros reserve extra link table entries which can be defined later with the RSCS DEFINE command. Also, the local linkid (local location ID) must be specified on the first GENLINK macro in the AXSLINKS file.

When you code the GENTAGQ macro for the TAGQUEUE COPY file, remember that the total number of tag slots must be equal to or greater than the number of holdslots defined either explicitly or implicitly in the AXSLINKS COPY file.

The following example shows the creation of the AXSLINKS, LAXLINES, and TAGQUEUE COPY files:

```
edit axslinks copy

NEW FILE:
EDIT:
input
INPUT:
* copy axslinks
  genlink id=mylocid,type=dmtnpt
  genlink id=newyork,class=a,keep=2,line=079,task=m1,type=dmtnpt
  genlink id=sanfran,class=a,keep=2,line=07a,task=m2,type=dmtsml
  genlink id=london,class=a,keep=4,line=07b,type=dmtsml
  genlink
  genlink
  genlink

EDIT:
file
R;
```

On the TYPE operand, DMTNPT refers to the Nonprogrammable Terminal (NPT) line driver program and DMTSML refers to the Spool MULTI-LEAVING (SML) line driver program.

```
edit laxlines copy
NEW FILE:
EDIT:
input
INPUT:
* copy laxlines
  genline line=079
  genline line=07a
  genline line=07b
  genline line=07c
  genline line=07d
  genline line=07e
  genline line=07f

EDIT:
file
R;

edit tagqueue copy
NEW FILE:
EDIT:
input
INPUT:
* copy tagqueue
  gentagq num=32

EDIT:
file
R;
```

STEP 6. CREATE DMTLOC MACRO LIBRARY

Using the COPY files created in Step 5, generate a CMS macro library called DMTLOC MACLIB, as follows:

```
maclib gen dmtloc axslinks laxlines tagqueue
```

If you must change your RSCS configuration at a later time, you have to change the COPY file and then generate a new DMTLOC macro library.

STEP 7. ASSEMBLE THE CONFIGURATION TABLE (DMTSYS)

Before you assemble the configuration table module, access the CP system disk (194) as an extension of the RSCS system disk.

```
access 194 b/a
DMSACC723I B (194) R/O
```

Now, assemble the RSCS configuration table module (DMTSYS) using the VMFASM EXEC procedure. Specify DMTR20 as the control file. This control file identifies DMTLOC as the macro library to be used during the assembly. The DMTR20 control file is supplied on the PLC tape and was loaded onto the 194 minidisk in Step 3. Invoke VMFASM as follows:

```
vmfasm dmtsys dmtr20
```

Note: If an error occurs due to an incorrectly coded macro, correct the macro and restart at Step 6.

STEP 8. INVOKE VMFBLD TO CREATE THE RSCS NUCLEUS

Access the 191 minidisk belonging to MAINT as the C-disk and invoke VMFBLD as follows:

```
access 191 c
vmfbld rscs build
```

The VMFBLD EXEC procedure prompts you with the message:

```
ENTER CP STAGING AREA DISK ADDRESS:
194
```

You respond with 194. The 194 minidisk contains all the RSCS files; they were loaded in Step 3.

VMFBLD next prompts you for the operands it needs to link to the RSCS system disk:

```
ENTER RSCS SYSTEM DISK LINK PARAMETERS:
USERID VADDR1 VADDR2
```

If you are following the examples in this procedure, you enter

```
rscs 191 195
```

The value you enter for vaddr2 must be the same value you specified for vaddr2 on the LINK command issued in Step 4; otherwise, the link is not allowed. The userid you specify is the userid of your RSCS virtual machine and the virtual address you specify for vaddr1 is the virtual device address of the RSCS system disk in the RSCS virtual machine.

VMFBLD then links to the RSCS system disk and copies four files (DMTNPT, DMTSML, DMTAXS, and DMTLAX) to it. You receive the following message:

```
TRANSFERRING 'RSCS' DISK RESIDENT TEXT...
```

Finally, VMFBLD builds and loads the RSCS nucleus. You receive the following messages:

```
DMTINI407R REWRITE THE NUCLEUS ? yes
DMTINI408R IPL DEVICE ADDRESS = 195
DMTINI409R NUCLEUS CYL ADDRESS = 003    (or, 004 for a 3330)
DMTINI410R ALSO IPL CYLINDER 0 (YES|NO) ? yes
```

For this example, respond as shown. You respond with the address of the RSCS system disk, in this case 195. You always respond with the same address you specified as vaddr2 when you were prompted for link parameters. The nucleus cylinder address depends on the device type of the RSCS system disk.

You receive the message

```
DMTAXS103E FILE 'spoolid' REJECTED -- INVALID DESTINATION ADDRESS
```

This message reflects the purging of the RSCS nucleus from the card reader.

At this time you have an RSCS system generated and written to the RSCS system disk, as indicated by the message:

```
DMTREX000I RSCS (VER 1, LEV 1, mm/dd/yy) READY
```

You can now log on the RSCS virtual machine, IPL the RSCS system disk and start your RSCS operations. See the VM/370: Remote Spooling Communications Subsystem (RSCS) User's Guide for information about using RSCS.

Part 4: Generating the 3704/3705 Control Program

If you do not want to support a 3704/3705 control program under VM/370 control, disregard Part 4.

Part 4 describes the procedures you must follow to generate, test, and run a 3704/3705 control program with VM/370. It includes the following information:

- Introduction
- Planning Considerations
- Generating and Loading the 3704/3705 Control Program
- Testing the 3704/3705 Control Program

You should generate a VM/370 system that supports the 3704/3705 first. "Part 1: Planning for System Generation" contains a section, "Generating a VM/370 System that Supports the 3704 and 3705," that tells you what you must include in your VM/370 system. When you have a VM/370 system that supports the 3704/3705, use the information in this part to generate and test a 3704/3705 control program that runs under VM/370 contol.

Introduction to the IBM 3704 and 3705 Communications Controllers

The IBM 3704 and 3705 Communications Controllers are programmable units. One of three programs can be generated to execute in 3704/3705 storage:

1. Network Control Program (NCP) performs many of the teleprocessing line-control and line-servicing functions previously performed by the central processing unit.

2. Emulation program (EP) permits existing teleprocessing systems, including VM/370, that use the IBM 2701, 2702, or 2703 Transmission Control Units or the Integrated Communications Adapter (ICA) of the System/370 Model 135, to execute without change on the 3704/3705.

3. Partitioned Emulation Program (PEP) allows the 3704/3705 to be divided so that both the NCP and EP can execute in one 3704/3705.

In this publication, the term "3704/3705 control program" refers to any of the three types of control programs: NCP, EP, or PEP.

VM/370 supports both the:

- IBM 3704 Communications Controller, Models A1-A4
- IBM 3705 Communications Controller, Models A1-D8

when attached to an IBM System/370 Model 135, 145, 155 II, 158, 165 II, or 168. Three terminals are supported: 1050, 2741, and CPT-TWX 33/35. The 3767 terminal (operating as a 2741) is not supported by lines in NCP mode. You can generate any of three kinds of 3704/3705 control programs (NCP, EP, or PEP) to execute under VM/370.

The minimum amount of 3704/3705 storage required for a NCP or PEP control program is 48K; the minimum required by an EP control program is 16K.

Planning Considerations

When planning for the installation of IBM 3704 and 3705 Communications Controllers, be sure that you are familiar with device characteristics, have the the appropriate publications and support package, and have a VM/370 system that supports the 3704/3705.

RELATED PUBLICATIONS

The Introduction to the IBM 3704 and 3705 Communications Controllers, Order No. GA27-3051, describes the general functions of the 3704 and 3705. It is a prerequisite publications for generating a 3704/3705 control program under VM/370.

If you are installing Version 2 of the Network Control Program/VS, the IBM 3704 and 3705 Communications Controllers, Network Control Program/VS Generation and Utilities Guide and Reference Manual (for OS/VS and DOS/VS TCAM Users), Order No. GC30-3007, is a corequisite publication. If you are installing Version 3 of the Network Control Program/VS, the IBM 3704 and 3705 Communications Controllers, Network Control Program/VS Generation and Utilities, Guide and Reference Manual (for OS/VS and DOS/VS VTAM Users), Order No. GC20-3008, is a corequisite publication. You must refer to one of these publications in order to code the 3704/3705 control program generation macros. Throughout Part 4, these publications are referred to as the 3704 and 3705 Generation and Utilities Guide.

3704 AND 3705 SUPPORT PACKAGE

Before you can generate a 3704/3705 control program, you must have one of the following OS/VS Network Control Program Support Packages. These are the only 3704/3705 support packages that contain the CMS file required for generating and loading 3704/3705 control programs under VM/370. The two support packages are:

- IBM 3704/3705 Network Control Program Support Package for OS/VS (Order No. 5744-BA1). VM/370 supports this package in network control, partitioned emulation, and emulation mode.
- IBM 3704/3705 Network Control Program Support Package for OS/VS (Order No. 5744-BA2). VM/370 supports this package in emulation mode only.

These packages contain the following basic material:

DOCUMENTATION

- A Program Directory
- IBM 3704 and 3705 Communications Controllers, Network Control Program/VS Generation and Utilities, Guide and Reference Manual (for OS/VS and DOS/VS TCAM Users), Order No. GC30-3007

--or--

- IBM 3704 and 3705 Communications Controllers, Network Control Program/VS Generation and Utilities, Guide and Reference Manual (for OS/VS and DOS/VS VTAM Users), Order No. GC30-3008
- IBM 3704 Control Panel Guide, Order No. GA27-3086.
- IBM 3705 Control Panel Guide, Order No. GA27-3087.

Machine Readable Material

- Magnetic tape containing the macros and modules of the 3704/3705 control program and the OS/VS system support programs.

VM/370 SUPPORT OF THE 3704 AND 3705

The IBM 3704/3705 Communications Controllers can support:

- Up to 352 low-speed start-stop lines
- Up to 60 medium-speed synchronous lines
- Line speeds from 45.2 baud to 50.0K baud
- Modem capability within the 3704/3705
- Limited-distance "hard-wire" capability
- 16K to 240K internal storage
- Remote 3277 and 3275 terminals with optional 3284, 3286, and 3288 printers

VM/370 supports all three versions of the 3704/3705 control programs:

- Emulation Program (EP)
- Network Control Program (NCP)
- Partitioned Emulation Program (PEP)

VM/370's support of the 3704/3705 does not include:

- Remote 3704/3705 Communications Controllers
- Binary synchronous terminals (However, the 2780 is supported when it is attached to an emulator mode line.)

EMULATION PROGRAM (EP) WITH VM/370

The EP 3704/3705 control program under VM/370:

- Emulates 2701, 2702, and 2703 operations
- Attaches to a System/370 byte multiplexer channel

- Supports up to 255 start-stop lines
- Supports up to 50 medium-speed synchronous lines

This support is equivalent to that provided in Release 1 of VM/370. The CP DIAL command and the TERMINAL APL ON and APL OFF command lines are supported. However, Release 2 of VM/370 provides additional support:

- Service programs and special CMS commands allow you to easily generate the EP control program in a CMS virtual machine.
- The CP NETWORK command allows you to load or dump the 3704/3705 and provides for automatic dumping and reloading if a fatal error occurs.

NETWORK CONTROL PROGRAM (NCP) WITH VM/370

The NCP 3704/3705 control program under VM/370 provides:

- A device-independent EBCDIC interface
- Communications line control handling
- Attachment to a System/370 byte multiplexer, block multiplexer, or selector channel
- Block and character checking
- Block and message buffering
- Assembly and disassembly of multiple-block transmissions
- Line error recovery procedures
- Checkpoint/restart switching to a backup processor
- User-written message processor routines

However, when an NCP 3704/3705 control program is under the control of VM/370:

- The CP DIAL command is not supported.
- The CP TERMINAL APL ON or APL OFF command line is not supported. If you issue the TERMINAL APL ON command at a terminal that is connected to VM/370 on a 3704/3705 line in NCP mode, you cannot execute your program and may have to IPL again to continue.
- NCP resources cannot be shared between VM/370 (the Control Program) and the virtual machines. A virtual 3704/3705 or 2701/2702/2703 is not supported.
- Special sign-on procedures are required (1) if you have the Multiple Terminal Access feature generated for your NCP control program and (2) if you have a 2741 terminal on an NCP-mode line. These procedures are described in "Step 11. Logging On Through the 3704/3705 Control Program" of the "Generating and Loading the 3704/3705 Control Program" section.

A VM/370 nucleus that supports the NCP version of the 3704/3705 control program is smaller than one that supports the EP or PEP

versions. Each line in NCP mode requires 48 bytes of free storage while each line in emulator mode requires 80 bytes of nucleus storage.

PARTITIONED EMULATION PROGRAM (PEP) WITH VM/370

The PEP 3704/3705 control program under VM/370:

- Combines the 2701/2702/2703 Emulation Program and the Network Control Program
- Allows for the concurrent use of NCP and emulator interfaces
- Provides for programmable switching of lines between NCP mode and emulator mode

If you execute a PEP control program under the control of VM/370, you should know that:

- The CP DIAL command is supported for lines generated as emulator lines and lines generated to vary between emulator mode and NCP mode. If the DIAL command is issued for a line that may be varied (and, if that line is currently in NCP mode), that line is automatically varied to emulator mode.
- The TERMINAL APL ON or APL OFF command line is supported only for lines currently in emulator mode.
- Only 3704/3705 lines in emulator mode may be dedicated.

Generating and Loading the 3704/3705 Control Program

Several commands and EXEC procedures generate and load the 3704/3705 control program. These commands and EXEC procedures execute in a CMS virtual machine. The commands are a part of the VM/370 system and are distributed with it.

A special version of the IBM 3704/3705 Network Control Program Support Package for OS/VS, Order No. 5744-BA1 for Version 2 and Order No. 5744-BA2 for Version 3, is available from PID for use under VM/370. These versions of the 3704/3705 package contains two CMS EXEC procedures for generating and loading the 3704/3705 control programs that are not available in the standard OS/VS 3704/3705 support package, Order No. 5744-AN1.

This section describes the step by step procedure for generating and loading the 3704/3705 control program. Each EXEC procedure and command is described as it is used. The action required at each step is summarized first and then explained in detail. The preceding "Planning Considerations" section, lists all the documentation, physical devices, programming, and other materials you need to have available before starting to generate the 3704/3705 control program.

STEP 1. LOG ON THE VM/370 SYSTEM

You must load a VM/370 system that supports a 3704/3705 control program.

Release 2 of VM/370 supports the NCP, EP, and PEP types of control programs. The system that you load also must have been generated with:

- The IBM 3704 or 3705 Communications Controllers specified on a RDEVICE system generation macro.
- The NAMENCP macro coded to create an entry in the VM/370 system name table (DMKSNT) for the 3704/3705 control program.
- Space reserved on a CP-owned volume to contain a copy of the 3704/3705 control program.

These VM/370 system generation requirements are described in Part 1.

STEP 2. SET UP A CMS VIRTUAL MACHINE

You must IPL a CMS virtual machine and be sure that the necessary devices are attached.

The 3704/3705 control program is generated using commands and EXEC procedures that execute in a CMS virtual machine. The CMS virtual machine must have the following resources:

- At least 1024K of virtual storage.
- One tape drive (9 track, 800 or 1600 bpi).

- Space available on the CMS A-disk (120 cylinders of a 3330 disk, all 203 cylinders of a 2314 disk, or 300 cylinders of a 3340 disk).

If the CMS virtual machine does not have these resources, use the CP DEFINE command to redefine the size of your virtual storage or send a message to the operator requesting him to attach the tape or disk device you need.

Be sure that there are no files on the A-disk with a filetype of COPY or TEXT. Use the CMS RENAME command to temporarily change such filetypes. A naming conflict can terminate the installation procedure for the distribution tape.

You need CP command privileges A, B, and G to install the 3704/3705 control program and, if you use the NETWORK TRACE command while testing the 3704/3705 control program, you need command privilege class F. Do not use class F unless you need it; for class F, I/O error recording is not done automatically. Check with the system administrator to ensure that your VM/370 directory entry has the appropriate command privileges.

STEP 3. LOAD THE IBM 3704/3705 CONTROL PROGRAM DISTRIBUTION TAPE FILES ONTO A CMS DISK

Use CMS commands to position the distribution tape at the proper file and to create CMS disk files from the tape files. The first file created from the tape files is an EXEC procedure that processes the rest of the tape files and creates the CMS disk files.

If you cannot mount the distribution tape yourself, send a message to the operator and have him mount the correct tape. The distribution tape contains six files. The fifth file is the LOADCC EXEC procedure which creates the necessary CMS files from the other tape files.

Use the CMS TAPE command to position the tape at the beginning of the fifth file:

```
tape fsf 4
```

Then, use the CMS TAPPDS command to create the LOADCC EXEC A1 file from the fifth file:

```
tappds * exec
```

If the file is successfully created, the response

```
FILE 'LOADCC EXEC A1' COPIED
```

appears on the terminal.

Invoke the LOADCC EXEC procedure to load all the necessary files and generate the 3705 Assembler:

```
loadcc
```

The LOADCC EXEC procedure creates the macro and text libraries, and generates the 3705 Assembler, that are needed to generate a 3704/3705 control program. The LOADCC EXEC procedure sends messages to the terminal to indicate its progress.

LOADCC issues the message

```
BUILD STAGE ONE MACLIB NCPGEN
```

and uses the second tape file to create the CMS file, NCPGEN MACLIB A1. It issues a second message

```
BUILD STAGE TWO MACLIBS NCPMAC01 NCPMAC02
```

and uses the third tape file to create the CMS files, NCPMAC01 MACLIB A1 and NCPMAC02 MACLIB A1. Using the fourth tape file, LOADCC creates the CMS file, OBJ3705 TXTLIB A1, after issuing the message

```
BUILD STAGE TWO TXTLIB OBJ3705
```

Finally, LOADCC issues the message

```
BUILD 3705 ASSEMBLER ASM3705
```

and creates two files, ARNGEND EXEC A1 and ASM3705 TXTLIB A1, from the sixth tape file. The ARNGEND EXEC procedure is invoked by LOADCC to generate the 3705 Assembler. The ASM3705 TXTLIB A1 file is used to generate the 3705 Assembler.

When the last message

```
END OF INSTALLATION
```

appears on the terminal, the distribution tape is no longer needed. At this time, the 3705 Assembler program, the macro libraries for the stage 1 and stage 2 generation procedures and the text library for the stage 2 generation procedure all exist on the CMS A-disk.

Note: You may find it helpful to dump the contents of the A-disk to tape at this time. If you save the tape dump, you have the pre-Stage 1 files. If errors are later encountered, you may need these files.

STEP 4. CODE THE 3704/3705 CONTROL PROGRAM MACRO INSTRUCTIONS

Code the 3704/3705 control program macro instructions and place them in a CMS file. Use the CMS Editor to create the file, which must have a filetype of ASM3705. VM/370 recommends that you assign the same filename to this CMS file as was specified previously in the NAMENCP macro. If the SAVE option is to be specified on the GEN3705 command, the filename must be the same. This ASM3705 file is used as input to Stage 1 of the 3704/3705 control program generation procedure.

Use the 3704 and 3705 Generation and Utilities Guide to code the macro instructions. Follow the macro instruction formats described in that publication except where suggestions and requirements are indicated in the following paragraphs.

Do not code the following macro instructions for an NCP 3704/3705 control program that executes under the control of VM/370:

BHSET	ENDBH
CLUSTER	IOLIST
COMP	LINELIST
DATETIME	SERVICE
DIALSET	STARTBH
EDIT	UBHR

The BUILD macro must be the first macro in the CMS file. Figure 23 lists the operands which VM/370 requires, recommends, and does not support. It also indicates (in parentheses) which of the three types of 3704/3705 control programs support each of the operands. For all other operands, refer to the 3704 and 3705 Generation and Utilities Guide.

Operand	Comments
LOADLIB=dsname OBJLIB=dsname	Required by the BUILD macro, but does not apply to VM/370. Specify a valid dsname. (NCP, PEP, and EP)
ABEND={YES / NO}	VM/370 recommends ABEND=YES be specified. (NCP and PEP only)
ANS={YES / NO}	VM/370 recommends ANS=NO be specified. (NCP and PEP only)
ASMXREF={YES / NO}	VM/370 recommends ASMXREF=YES be specified. (NCP, PEP, and EP)
BFRS={SIZE / 60}	VM/370 recommends BFRS=68 be specified. (NCP and PEP only)
CHKPT={YES / NO} CSMHDR=chars CSMHDRC=chars CSMMSGC=chars	VM/370 ignores these operands. (NCP and PEP only)
CUID=chars DIALTO={count / 60.0}	VM/370 does not support these operands. (NCP and PEP only)
ERASE={YES / NO}	VM/370 recommends ERASE=YES be specified. (NCP AND PEP only)
ITEXTO={COUNT / NONE}	VM/370 requires the default value. (NCP and PEP only)
JOBCARD={YES / NO / MULTI}	VM/370 recommends JOBCARD=MULTI. (NCP, PEP, and EP)
MTARTRY={COUNT / _0}	VM/370 recommends MTARTRY=16 be specified. (NCP and PEP only)

Figure 23. BUILD Macro Operands for VM/370 (Part 1 of 2)

Operand	Comments
NEWNAME={NCP001 / PEP001 / symbol}	VM/370 requires that the value of NEWNAME be the same as the name previously specified in the NAMENCP macro and the name that subsequently will be specified in the SAVENCP command. Also, if the GEN3705 command is to be issued with the SAVE option, the value of NEWNAME must be the same as the "fname" specified on the GEN3705 command. (NCP, PEP, and EP)
OLT={YES / NO}	VM/370 recommends OLT=NO be specified. (NCP and PEP only)
PNLTEST={YES / NO}	VM/370 recommends PNLTEST=YES be specified. (NCP, PEP, and EP)
QUALIFY={symbol / NONE / SYS1}	VM/370 requires the default value. (NCP, PEP, and EP)
TRACE={[NO / YES][,{SIZE / 10}]}	VM/370 recommends that TRACE=YES be specified. (NCP, PEP, and EP)
UT1=dsname UT2=dsname UT3=dsname	VM/370 ignores these operands. (NCP, PEP, and EP)
XBREAK={integer / NONE}	VM/370 requires that XBREAK be specified with a minimum value of 3. (NCP and PEP only)
XITB={YES / NO}	VM/370 does not support this operand. (NCP, PEP, and EP)

Figure 23. BUILD Macro Operands for VM/370 (Part 2 of 2)

CSB MACRO INSTRUCTION

The CSB macro instruction is required. See the 3704 and 3705 Generation and Utilities Guide for more information about coding the CSB macro instruction.

SYSCNTRL MACRO INSTRUCTION

The SYSCNTRL macro instruction specifies which of the dynamic control features are to be included in the 3704/3705 control program. SYSCNTRL

applies to only NCP and PEP 3704/3705 control programs. The minimum options required by VM/370 are:

```
SYSCNTRL       OPTIONS=(LTRACE,
                        STORDSP,
                        MODE,
                        RCOND,
                        RIMM,
                        RECMD,
                        RDEVQ,
                        RCNTRL,
                        DEACTDV,
                        ACTDV)
```

The other options for SYSCNTRL listed in the 3704 and 3705 Generation and Utilities Guide may be specified; however, they are not recognized

by VM/370 and use storage unnecessarily. The DEACTDV (deactivate device) and ACTDV (activate device) options are not listed in the 3704 and 3705 Generation and Utilities Guide. Nevertheless, they are valid and must be specified for VM/370.

HOST MACRO INSTRUCTION

One HOST macro is required. It applies to only NCP and PEP 3704/3705 control programs. For VM/370, it must have the following operands with the values shown:

```
HOST          INBFRS=4,
              DELAY=0,
              MAXBFRU=6,
              UNITSZ=96,
              BFRPAD=(34,34),
              TIMEOUT=NONE
```

For a PEP 3704/3705 control program, the "NCPCHAN=channel address" operand must also be specified.

THE MACRO INSTRUCTIONS FOR THE MULTIPLE TERMINAL ACCESS (MTA) FEATURE: MTALCST, MTALIST, MTAPOLL, MTATABL, AND GROUP

VM/370 recommends that you use the NCP Multiple Terminal Access (MTA) feature. This feature allows the 3704/3705 control program to accept switched-line connections for any of several terminal types, on the same physical line interface. The 3704/3705 control program determines the terminal line code and line speed when the switched connection is initially established, and passes the terminal type to the host access method.

The MTA feature of the NCP supports 1050, 2741, CPT-TWX, 2740-1, and 2740-2; however, VM/370 supports only 1050, 2741, and CPT-TWX 33/35 terminals. Therefore, do not specify the 2740-1 or 2740-2 devices on any of the MTA macros for a 3704/3705 control program that executes under VM/370. The 3704 and 3705 Generation and Utilities Guide. contains all the information you need to code the MTA macro; however, use Figure 24 when coding the LCTYPE operands.

Macro	Cperand	Comments
MTALCST	LCTYPE={1050 / 2741 / TWX}	VM/370 does not support 2740A, 2740D, 2740E, or 2740F specificatcns fcr the LCTYPE operand.
MTALIST	LCTYPE=(type,...) where: type may be 1050, 2741, or TWX.	
MTATABL	LCTYPE={1050 / 2741 / TWX}	

Figure 24. LCTYPE Operands for MTA Macros with VM/370

Figure 25 shows valid MTA macros for a 3704/3705 control program to be executed with VM/370. The first set of macros (labeled MASTER through MTELE) describes an actual hardware configuration. Along with this set of macros, it is necessary to specify TERM=MTA on the GROUP, LINE, and TERMINAL macros; and to specify MTALIST=MASTER on each LINE macro to be used with MTA. The second set of macros (labeled FAKE2741, FAKE1050, and FAKETTY) describes a software configuration, a configuration with no lines.

MTA Macros			Comments
MASTER	MTALIST	LCTYPE=(2741,TWX,1050)	Master selection list
M2741E	MTALCST	GROUP=FAKE2741,SPEED=134, LCTYPE=2741,CLOCKNG=INT, CODE=EBCD	PTTC/EBCD 2741
M2741C	MTALCST	GROUP=FAKE2741,SPEED=134, LCTYPE=2741,CLOCKNG=INT, CODE=COR	Correspondence 2741
M1050E	MTALCST	GROUP=FAKE1050,SPEED=134, LCTYPE=1050,CLOCKNG=INT, CODE=EBCD	Standard 1050
MTELE	MTALCST	GROUP=FAKETTY,SPEED=110, LCTYPE=TWX,CLOCKNG=INT	CPT-TWX 33/35
	MTATABL	LCST=M2741E,CODE=EBCD,LCTYPE=2741	
	MTATABL	LCST=M2741C,CODE=COR,LCTYPE=2741	
	MTATABL	LCST=M1050E,CODE=EBCD,LCTYPE=1050	
	MTATABL	LCST=MTELE,LCTYPE=TWX	
	MTAPOLL	POLL=(61F0)	1050 common polling characteristics
FAKE2741	GROUP	DIAL=YES,LNCTL=SS,TYPE=NCP,TERM=2741	
FAKE1050	GROUP	DIAL=YES,LNCTL=SS,TYPE=NCP,TERM=1050, POLLED=YES	
FAKETTY	GROUP	DIAL=YES,LNCTL=SS,TYPE=NCP,TERM=TWX	

Figure 25. Example of MTA Macros for Use with VM/370

If, for multiple terminal access support, it is necessary to define a standalone line group (a GROUP macro followed by no LINE and TERMINAL macros), specify the following operands in the GROUP MACRO:

DIAL=YES
LNCTL=SS
TERM=1050 (TWX or 2741)
POLLED=YES (only if you specify TERM=1050)

All other LINE and TERMINAL operands are invalid for a standalone line group and, if coded, are ignored.

GROUP, LINE, AND TERMINAL MACRO INSTRUCTIONS

These three macros describe the physical and logical configuration of the communications network accessed through the 3704/3705 control program. Since VM/370 does not support either multi-drop lines or cluster control units, the 3704/3705 configuration for VM/370 is generally simple, with only one GROUP macro for each communications scanner and one TERMINAL macro for each LINE macro. For VM/370, it is often easiest to specify most of the operands of the TERMINAL and LINE macros on the GROUP macro. The 3704 and 3705 Generation and Utilities Guide describes the GROUP, LINE, and TERMINAL macro instructions in detail and lists all the operands of the configuration macros, telling you where each operand is described and also where it may be coded.

VM/370 requires the ATTN, MONITOR, DUPLEX, and FEATURE operands. These operands allow VM/370 to detect and respond to a terminal attention interrupt and to recognize when a data set has been hung up. The CRDLAY operand is recommended because it prevents typing when the terminal carriage is returned. POLLED=YES is required for any 1050 device. For the GROUP macro, VM/370 requires the default value for the REPLYTO operands and recommends the default value for the TEXTTO operand. The macros shown in Figure 26 describe a sample Network Control Program configuration without PEP support.

Configuration	Macros	Comments
FIRST GROUP	DIAL=YES,	
	LNCTL=SS,	
	SPEED=134,	
	MTALIST=MASTER,	MTA in use, see Figure 25
	TERM=MTA,	
	CRDLAY=YES,	Prevents typing while carriage
	CDATA=YES,	is in motion
	ATTN=ENABLED,	Required
	DUPLEX=FULL,	Required
	FEATURE=(ATTN,BREAK),	Required
	MONITOR=YES,	Required
	POLLED=YES	Required for 1050
LINE20 LINE	ADDRESS=020	First line, base module
TERM20 TERMINAL	TERM=MTA,CTERM=YES	Logical connection base
LINE21 LINE	ADDRESS=021	
TERM21 TERMINAL	TERM=MTA,CTERM=YES	

Figure 26. Example of Configuration Macros for Use with VM/370

Although the SPEED operand specifies 134.5 bps, the MTA feature is able to recognize a CPT-TWX terminal, and has the ability to change the oscillator rate to the required 110 bps. However, the MTA feature cannot be used to intermix CPT-TWX with 1050 or 2741 terminals unless the hardware communications scanner(s) have both the 134.5 bps oscillator and the 110 bps oscillator.

GENEND MACRO INSTRUCTION

The GENEND macro indicates the end of the 3704/3705 macro input file. It must be the last record in the CMS file you are building as input to Stage 1.

SPECIAL MACRO CODING CONSIDERATIONS FOR THE PARTITIONED EMULATION PROGRAM (PEP)

The PEP control program combines the 2701/2702/2703 Emulation Program with the NCP. In some cases, both the emulator and the NCP macro operands must be specified in the Stage 1 input file. Code all the macros as though you were generating an NCP control program. Then, add the additional emulator operands which are required. For the BUILD macro, the emulator operands HICHAN, LOCBAN, and TYPGEN=PEP are required. For the HOST macro, an additional operand, NCPCBAN, is required to specify which subchannel address is to be used for communication with the NCP portion of PEP. The SYSCNTRL macro is unchanged.

The MTA facility can be used with only the NCP; and only with NCP resources that cannot be varied to emulation mode. Whenever MTA operands are specified for lines that can be varied, errors result in the Stage 1 assembly.

SPECIAL MACRO CODING CONSIDERATIONS FOR THE EMULATION PROGRAM (EP)

There are no strict dependencies between the host access method and the emulation program; consequently, few guidelines are necessary for an emulation program generation. However, be careful when configuring emulator lines for CPT-TWX terminals. While VM/370 normally accepts incoming calls from either 1050 or 2741 terminals on the same physical line, that same line cannot be used for CPT-TWX terminals. When generating the VM/370 system, ensure that the hardware configuration specified in the CP module DMKRIO matches the configuration of the Emulation Program for CPT-TWX lines; the exact configuration of 1050 and 2741 lines is not critical.

Note: The base address of the 3704/3705 (the address used tc load and/or dump the control program) can never be specified for use as a telecommunications line. VM/370 treats the base address as a separate entity for use only during the load and dump operation.

STEP 5. DEFINE THE MACRO AND TEXT LIBRARIES

The macro and text libraries created from the distribution tape in Step 3 must be made available to CMS. One macro library (NCPGEN) is needed for the Stage 1 generation procedure and two macro libraries (NCPMAC01 and NCPMAC02) and one text library (OBJ3705) are needed for the Stage 2 generation procedure. It is easiest to define all the libraries before starting Stage 1. Use the CMS GLOBAL command:

```
GLOBAL MACLIB NCPGEN NCPMAC01 NCPMAC02
GLOBAL TXTLIB OBJ3705
```

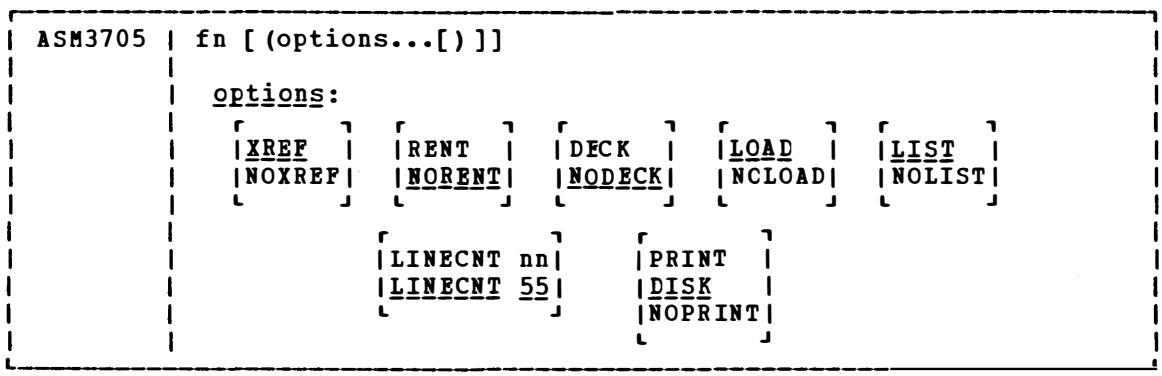
STEP 6. THE STAGE 1 GENERATION PROCEDURE

The Stage 1 generation procedure accepts the CMS file you created in Step 4 as input and produces the Stage 2 input file that is needed in Step 7.

The Stage 1 generation procedure is performed by invoking the 3705 Assembler to process the 3704/3705 control program macro instructions. It produces one file with the same filename as the input file and with a filetype of TEXT. This TEXT file contains 3705 Assembler source statements and job control language (JCL) statements.

THE ASM3705 COMMAND

Use the CMS ASM3705 command to invoke the 3705 Assembler to assemble the macro instruction file. The 3705 Assembler processing and output are controlled by the options selected. The format of the ASM3705 command is:



where:

fn specifies the filename of the source file to be assembled. This source file contains the 3704/3705 control program macro instructions. The file must have a filetype of ASM3705 and fixed-length, 80-character records.

Options

If duplicate or conflicting options are specified, the last one entered in the command line is in effect.

XREF includes a cross-reference symbol table in the LISTING file.

NOXREF suppresses the cross-reference symbol table.

RENT checks the source file to see if it satisfies reentrancy requirements.

NORENT suppresses the check for satisfaction of reentrancy requirements.

DECK spools the output object module, fn TEXT, to the punch.

NODECK suppresses the spooling of the output object module, fn TEXT, to the punch.

LOAD creates a TEXT file on disk for the program that was assembled.

NOLOAD suppresses the creation of a TEXT file on disk for the program that was assembled.

LIST produces a LISTING file.

NCLIST produces no LISTING file.

PRINT spools the LISTING file to the printer.

DISK puts the LISTING file on disk.

NCPRINT produces no LISTING file.

LINECNT nn
specifies the number of lines per output printer page. A default of 55 lines is assumed.

Files Created by the ASM3705 Command

TEMPORARY WORKFILES: Three files are temporarily created for each assembly:

```
fn SYSUT1
fn SYSUT2
fn SYSUT3
```

Any existing files with the same file identifiers are erased at the beginning of the assembly. These files are placed on the read/write disk with the most available space. Work space is automatically allocated as needed during the assembly and returned to available status when the assembly is complete. Insufficient space causes abnormal termination of the assembly.

PERMANENT FILES: One or two permanent files may be created during a successful assembly:

```
fn TEXT
fn LISTING
```

The fn TEXT file contains the output object module if the LOAD option is in effect. The fn LISTING file contains a listing of source statements, assembled machine code, and other associated information based on the options selected. This file is created unless the NOPRINT or NOLIST options are selected. The LISTING and TEXT files are placed on (1) the disk from which the source file was read, (2) its parent or (3) the primary disk, disk, unless you created a file definition for these files placing them on a non-DASD device. Failure to obtain sufficient space for these files results in abnormal termination of the assembly.

SPECIAL CONSIDERATIONS FOR THE STAGE 1 ASSEMBLY

The Stage 1 assembly can be very lengthy. The amount of time the Assembler takes depends upon the macro options selected and the number of users on the VM/370 system. Do not be surprised if the assembly for an NCP control program takes over an hour.

The LISTING file produced by the Stage 1 assembly is quite large. If you let the ASM3705 command option default to DISK, much of the space on your A-disk is used. Therefore, VM/370 recommends that you specify the PRINT option when you issue the ASM3705 command. Also, there are many

macro expansions that make the LISTING file larger. VM/370 recommends that you insert a 'PRINT NOGEN' assembler statement in front of the first macro instruction in the input file to suppress the printing of the macro expansions and reduce the size of the LISTING file.

You should examine the output of the Stage 1 assembly carefully and produce a list of resource IDs, with their characteristics, for the operations personnel. The cross-reference list for operations should include:

- Resource ID
- Type of resource (line or terminal)
- Type of line (NCP-mode, EP-mode, or variable)
- Location

STEP 7. THE STAGE 2 GENERATION PROCEDURE

During the Stage 2 generation procedure the TEXT file produced in Step 6 is scanned. That TEXT file contains several job steps of 3705 Assembler source statements with embedded OS JCL statements. The JCL statements are removed and a unique CMS 3705 Assembler source file is created for each job step in the input file. An EXEC procedure is also created to assemble and link edit the source files. When the EXEC procedure is invoked, it produces the load module file (and, optionally, saves a copy of the control program in page-format on a CP-owned volume).

THE GEN3705 COMMAND

Use the CMS GEN3705 command to invoke the Stage 2 service program. Command options let you determine whether or not GEN3705 includes a command in the EXEC procedure to save a copy of the load module on disk, or if GEN3705 invokes the EXEC procedure automatically. The format of the GEN3705 command is:

```
┌──────────────────────────────────────────────────────────────────────┐
| GEN3705 | fname ftype [fmode] [(options...[)]]                        |
|         |                                                             |
|         | Options:                                                    |
├         | ┌     ┐  ┌      ┐                                          |
|         | |RUN  |  |SAVE  |                                          |
|         | |NORUN|  |NOSAVE|                                          |
|         | └     ┘  └      ┘                                          |
└──────────────────────────────────────────────────────────────────────┘
```

where:

fname specifies the filename of the Stage 2 input stream produced by the Stage 1 assembly. The file must contain fixed-length, 80-character records.

ftype specifies the filetype of the Stage 2 input stream. The filetype is normally TEXT.

fmode specifies the filemode.

Options:

If duplicate or conflicting options are specified, the last option entered on the command line is in effect.

RUN causes the output EXEC file to be executed at the conclusion of the GEN3705 processing.

NORUN suppresses the execution of the output EXEC file.

SAVE includes a SAVENCP command in the output EXEC file to create a page-format copy of the 3704/3705 control program on a VM/370 CP-owned volume.

NOSAVE does not include the SAVENCP command in the output EXEC file.

Files Created by the GEN3705 Command

Three types of permanent files are created when the GEN3705 command successfully executes: ASM3705, TEXT, and EXEC files.

```
fname00  ASM3705
fname01  ASM3705
   .        .
   .        .
   .        .
fnamenn  ASM3705


fnameL0  TEXT
fnameL1  TEXT
   .       .
   .       .
   .       .
fnameLn  TEXT


fname    EXEC
```

A separate ASM3705 file is created for each assembly job step in the Stage 2 input file. Each ASM3705 file created by GEN3705 is given a unique filename of the form 'fnamenn'. The first six characters of the input filename are concatenated with a two-digit number. For example, if the input file is NCP320 TEXT, the output files are NCP32000 ASM3705, NCP32001 ASM3705, ..., NCP320nn ASM3705. These files are used as input to the 3705 Assembler when it is invoked by the Stage 2 EXEC procedure.

The GEN3705 program creates several TEXT files. These files contain only linkage editor control statements, those statements necessary to build the load module file for the 3704/3705 control program. Each of the TEXT files created is given a unique filename of the form 'fnameLn'. The first six characters of the input filename are concatenated with the letter L and a one-digit number. For example, if the input file is NCP320 TEXT, the linkage editor output files are NCP320L0 TEXT, NCP320L1 TEXT, ..., NCP320Ln TEXT.

The filenames assigned to the linkage editor and assembler files must be different. If the filenames were the same, when the ASM3705 files are later assembled, TEXT files would be produced which would have file identifiers that conflict with the linkage editor files.

The EXEC macro file created contains the CMS commands necessary to invoke the ASM3705 command for each of the ASM3705 files, and to subsequently invoke the linkage editor for each of the Assembler TEXT files. If the SAVE option is specified, the EXEC file also contains the SAVENCP command which loads the 3704/3705 control program image into virtual storage and creates the page-format copy of it on a CP-owned volume. The filename of the Stage 2 input file is used as the 'ncpname' operand for the SAVENCP command.

SPECIAL CONSIDERATIONS FOR THE STAGE 2 GENERATION PROCEDURE

VM/370 recommends that you specify the RUN option. When the RUN option is specified, GEN3705 stacks a CMS command line to cause the EXEC file to execute following the completion of the GEN3705 program. This technique minimizes the virtual storage overhead during the EXEC file execution.

If you do not specify the SAVE option, you have to explicitly issue the SAVENCP command. If you do specify the SAVE option, be sure that the input file has the same filename as the entry reserved in the system name table. The system name table is created when a NAMENCP macro is issued during a VM/370 system generation. The NAMENCP macro is described in "Part 2: Defining Your VM/370 System" and the building of the system name table is described in "Part 3: Generating VM/370 (CP, CMS, and RSCS)."

STEP 8. INVOKE THE EXEC PROCEDURE CREATED IN STEP 7

If you specified RUN on the GEN3705 command, this step is executed for you. If you did not specify RUN on the GEN3705 command, you must invoke the EXEC procedure that the GEN3705 program created.

The EXEC procedure is given the same filename as the GEN3705 input file. It is invoked by entering that filename at the terminal. For example, if the input file is NCP320 TEXT, the EXEC file is named NCP320 EXEC, and can be invoked by issuing:

```
NCP320
```

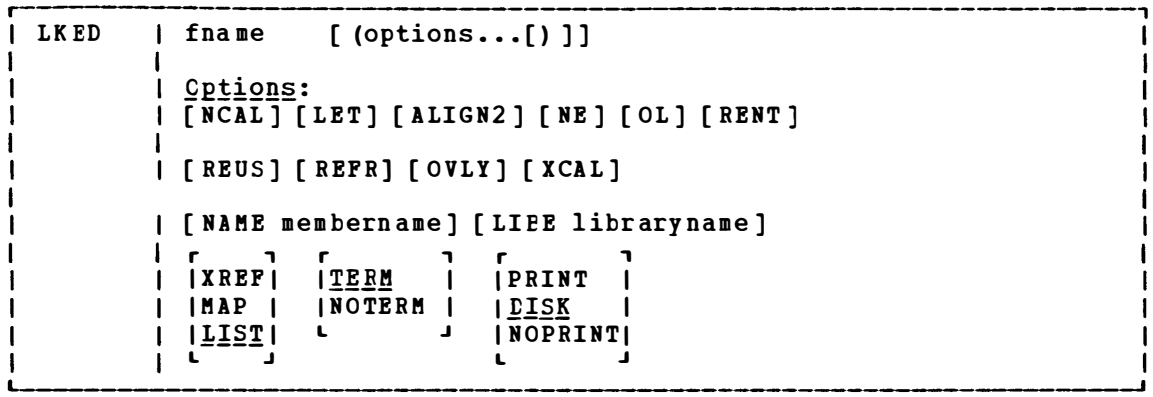
at the terminal.

This EXEC procedure contains CMS commands that:

- Assemble the 3705 source files (ASM3705 commands).
- Build the TXTLIB which the 3705 Assembler needs (TXTLIB commands).
- Define all the necessary files; such as, the SYSLIB and SYSLMOD files, load libraries, and text libraries (FILEDEF commands).
- Link edit the 3705 text files creating a load module (LKED commands).

You do not have to issue the ASM3705 and LKED commands that create the 3704/3705 control program load module; the EXEC procedure does that for you. The ASM3705 command is described in Step 6. The FILEDEF and TXTLIB commands are described in the *VM/370: Command Language Guide for General Users*.

Use the CMS LKED command to create the 3704/3705 control program load module from the 3705 Assembler object files. The format cf the LKED command is:



where:

fname specifies the filename of the object file to be processed. The file must have a filetype of TEXT and fixed-length, 80-character records.

Options:

If duplicate or conflicting linkage editor options are specified, the resolution is performed by the linkage editor in accordance with its normal procedures. If duplicate or conflicting CMS-related options are specified, the last one entered on the command line is in effect. The CMS-related options are: TERM, NOTERM, PRINT, DISK, NCPRINT, NAME, and LIBE.

NCAL suppresses the automatic library call function of the linkage editor.

LET suppresses marking of the load module "not executable" in the event of some linkage editor error condition.

ALIGN2 indicates that boundary alignment specified in the linkage editor input file is to be performed on the basis of 2048-byte boundaries. If this option is omitted, alignment is performed on the basis of 4096-byte boundaries.

NE marks the load module output as "not to be edited" such that it cannot be processed again by the linkage editor.

OL marks the load module output "only loadable".

RENT marks the load module reenterable.

REUS marks the load module reuseable.

REFR marks the load module refreshable.

OVLY processes an overlay structure.

XCAL allows valid exclusive CALLs in the overlay structure.

NAME membername
is the member name to be used for the load module created. The member name specified here overrides the default name, but it cannot override a name specified via the linkage editor NAME control statement.

LIBE libraryname
is the filename of a LOADLIB file where the output load module is to be placed. The LOADLIB file specified here may also be used for auxiliary input to the linkage editor via the INCLUDE statement.

XREF produces an external symbol cross reference for the modules being processed.

MAP produces only a module map for the processed module(s).

LIST includes only linkage editor control messages in the printed output file.

TERM displays any linkage editor diagnostic messages at the user terminal.

NCTERM suppresses the displaying of diagnostic messages.

PRINT spools the linkage editor printed output file to the printer.

DISK stores the linkage editor output in a CMS disk file with a filetype of LKEDIT.

NOPRINT produces no output file.

Linkage Editor Control Statements

Only a subset of the possible linkage editor control statements are meaningful in CMS. Since the CMS interface program is not able to examine the input data for the LKED command, all of the control statements are allowed, even though several of them result in the creation of a load module file which cannot be used under CMS. For both command options and control statements, see the publication OS/VS Linkage Editor and Loader.

Files Created by the LKED Command

Temporary Workfile: The LKED command produces one temporary file:

```
fname SYSUT1
```

This file is temporarily created for each link-edit step; any existing file with the same file identifier is erased at the beginning of the link edit. This file is placed on the read/write disk with the most available space. Work space is automatically allocated as needed during the link edit and returned to available status when the link edit is

complete. Insufficient space causes abnormal termination of the link edit.

Permanent Files: The LKED command produces two permanent files:

```
fname LCADLIB
fname LKEDIT
```

The 'fname LCADLIB' file contains the load module(s) created by the linkage editor. This file is in CMS simulated partitioned data set format, as created by the CMS OS data management macros. The filename of the input file becomes the filename of the LOADLIB file, unless the LIBE option is specified. The filename of the input file also becomes the member name of the output load module, unless either the NAME option or a NAME control statement is used. One or more load mcdules may be created during a single LKED command execution if the NAME linkage editor control statement is used in the input file. When the NAME control statement is used, that name becomes the member name in the LOADLIB file. The replace option of the NAME statement determines whether existing members with the same name are replaced or retained.

The 'fname LKEDIT' file contains the printed output listing produced according to the XREF, MAP, or LIST options. This file is created on disk unless the PRINT or NOPRINT option is specified. The LOADLIB and LKEDIT files are placed on (1) the disk from which the input file was read, (2) the parent disk, or (3) the primary disk. Failure to obtain sufficient space for these files results in abnormal termination of the linkage editor.

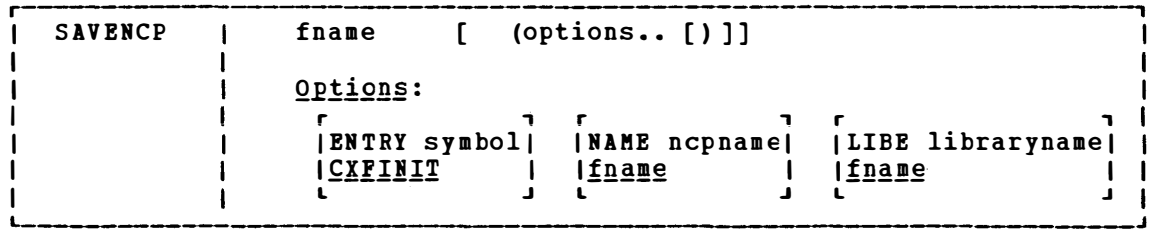
STEP 9. SAVE THE 3704/3705 CONTROL PROGRAM IMAGE ON DISK

If you specified SAVE on the GEN3705 command, this step is executed for you. If you did not specify SAVE on the GEN3705 command, you must issue the SAVENCP command yourself.

Note: The VM/370 command privilege class A, B, or C is required to use the SAVENCP command.

THE SAVENCP COMMAND

Use the CMS SAVENCP command to read a 3704/3705 control program load module created by the LKED command, and to load it into virtual storage in the CMS user area. Once the load is performed, SAVENCP scans the control program image and extracts the control information required by CP. The control information is accumulated in one or more 4096-byte pages in the CMS user area. When all of the necessary control information is extracted, SAVENCP builds the Communications Controllers Parameter List (CCPARM) and issues the DIAGNOSE X'50' instruction to create the page-format copy of the control program on a CP-owned volume. The format of the SAVENCP command is:



where:

fname is the filename of the LOADLIB file where the 3704/3705 control program load module resides; unless LIBE is specified, in which case, it specifies the member name of the image within the LOADLIB. This name is used as the ncpname for the DIAGNOSE instruction, unless the NAME option is also specified.

Options

ENTRY symbol
is the external symbol of the entry point in the 3704/3705 control program load module. The default, CXFINIT, is the entry point of the standard NCP or PEP control programs. (The standard entry for the Emulation Program is CYASTART.) If the SAVE option of the GEN3705 command is specified, this symbol is set in the output EXEC file according to the Stage 2 input file.

NAME ncpname
is the ncpname to be used when the DIAGNOSE parameter list is built. The ncpname specified must match an entry in the system name table. These entries are created with the NAMENCP macro when VM/370 is generated.

LIBE libraryname
is the filename of a load module library file, filetype LOADLIB, which contains the control program image as member 'fname'.

EXECUTION OF THE SAVENCP PROGRAM

The DIAGNOSE X'50' instruction invokes the CP module DMKSNC to:

- Interpret the parameter list (CCPARM) built by SAVENCP.
- Check the parameter specifications against the NAMENCP macro for the 3704/3705 control program.
- Write the page-format image of the control program onto the appropriate CP-owned volume.

The parameter list for the DIAGNOSE instruction must start on a 4096-byte boundary.

When the DIAGNOSE X'50' instruction is executed, the module DMKSNC searches the DMKSNT module for a NAMENCP macro of the same ncpname as the one in the CCPARM parameter list. The values specified in the parameter list are compared to those specified in the NAMENCP macro. If

any parameters conflict, an error message is displayed at the terminal. If no error conditions are detected, DMKSNC starts to transfer the control program image from CMS virtual storage to the CP-owned volume specified in the NAMENCP macro. Successful completion of this process completes the generation of a 3704/3705 control program for VM/370 use.

STEP 10. LOAD THE 3704/3705 CONTROL PROGRAM

The 3704/3705 control program is automatically loaded each time the VM/370 system is loaded, if the CPNAME operand was specified on the RDEVICE macro when VM/370 was generated and if the 3704/3705 is online. If the CPNAME operand was not coded, you must issue the CP NETWORK LOAD command line to load a 3704/3705 control program into the 3704/3705 Communications Controllers' storage.

THE NETWORK LOAD COMMAND LINE

Use the NETWORK LOAD command to initiate the loading of an NCP, PEP, or EP control program into a 3704/3705 Communications Controller. The format of the NETWORK LOAD command line is:

```
┌──────────────────────────────────────────────────────────────────────┐
│ NETwork   │ LOAD raddr ncpname                                       │
└──────────────────────────────────────────────────────────────────────┘
```

where:

LOAD initiates the control program load operation.

raddr is the real address of the 3704/3705 to be loaded.

ncpname is the name, defined by a NAMENCP macro, of the 3704/3705 control program image to be loaded into the 3704/3705 specified by raddr.

EXECUTION OF THE NETWORK LOAD COMMAND

The NETWORK LOAD command accesses the control program image using the information in the system name table (DMKSNT) entry created by the NAMENCP macro. If the 3704/3705 specified in the command is not in an "IPL Required" state at the time the command is issued, the message:

```
DMKNET461R CTLR raddr IPL NOT REQUIRED; ENTER "YES" TO CONTINUE:
```

appears at the terminal. If the reply to the message is other than "yes", the command terminates without loading the 3704/3705. Otherwise, the loader bootstrap routines are written to the 3704/3705 and loading starts. VM/370 does not execute the "bring-up" test routines as a part of the load process. If these tests are to be made, they must be run from a virtual machine with the 3704/3705 dedicated.

When the load of the control program image is complete, the command processor verifies that the 3704/3705 configuration described by the control program can be serviced by the VM/370 CP control blocks in

storage. In the case of CPTYPE=NCP (or PEP), this involves creating (or refreshing) the control list associated with the 3704/3705 real device block (RDEVBLOK). The information necessary to do this is contained in the system pages at the beginning of the control program image on secondary storage.

SPECIAL CONSIDERATIONS FOR LOADING THE EP 3704/3705 CONTROL PROGRAM

If a 3704/3705 Emulation Program is automatically reloaded after a 3704/3705 failure, the system may loop after the restart. The message

```
DMKRNH463I CTLR raddr UNIT CHECK; RESTART IN PROGRESS
```

and two responses

```
CTLR raddr DUMP COMPLETE
CTLR raddr ncpname LOAD COMPLETE
```

indicate that the 3704/3705 has been reloaded. If the system loops after the second response, you must reset all emulator lines from the 3704/3705 control panel.

If the automatic dump feature is not enabled, one of the messages

```
DMKRNH462I CTLR raddr UNIT CHECK; IPL REQUIRED

-- or --

DMKRNH464I CTLR 'raddr' CC=3; DEPRESS 370X "LOAD" BUTTON
```

indicates a 3704/3705 abnormal termination. The 3704/3705 Emulation Program must be reloaded via the NETWORK LOAD command. If the system loops when an attempt is made to enable the lines, you must reset all emulator lines from the 3704/3705 control panel.

The IBM 3704 and 3705 Communications Controllers Operator's Guide describes the procedure for resetting emulator lines from the 3704/3705 control panel in its "Generating Channel End/Device End with Emulator Program" section.

SPECIAL CONSIDERATIONS FOR LOADING THE NCP AND PEP 3704/3705 CONTROL PROGRAMS

Special care must be taken when loading a Network Control Program or Partitioned Emulation Program. The NETWORK LOAD command should not be used to load either an NCP or PEP except at VM/370 system IPL time, unless that same 3704/3705 control program was active just prior to the load (that is, unless it is reloaded immediately). A VM/370 system abnormal termination (code PTR007) may result if the 3704/3705 is loaded, for the first time, during normal operation with an NCP or PEP program. However, if an NCP or PEP 3704/3705 control program must be loaded during system operation, all resources must first be freed.

If there are active resources, the resources must be disabled and the NETWORK SHUTDOWN command must be issued before the operator can successfully issue the NETWORK LOAD command.

STEP 11. LOGGING ON THROUGH THE 3704/3705

Because a 3704/3705 can support emulator-mode lines, NCP-mode lines, and lines that can be varied to either mode, and can also suppcrt a variety of terminals, the procedure for logging on is sometimes complicated. Use the following procedure to log on to VM/370.

TURN THE POWER ON

First, turn the power on for your terminal and wait 15 to 30 seconds.

CHECK FOR AN ONLINE MESSAGE

Second, look for an online message at your terminal.

If one of the following messages appears at your terminal

```
vm/370 online      xxxxxx xxxxxx

           -- or --

xxxxxx xxxxxx      vm/370 online
```

your terminal is a 2741 connected to VM/370 via a 2701/2702/2703 line or via a 3704/3705 line in emulation mode. You can proceed with the normal logon procedure for your type of terminal, as described in the VM/370: Terminal User's Guide.

If the messsage

```
vm/370 online
```

appears at your terminal, your terminal is:

- A 2741 connected to VM/370 via a 3704/3705 line in NCP mode, without the Multiple Terminal Access feature, or
- A 1050 or CPT-TWX Model 33/35 terminal connected to VM/370 in EP mode.

You can proceed with the normal logon procedure for your terminal type. This procedure is described in the VM/370: Terminal User's Guide.

If you receive a message at your terminal in the form

```
xxxxxx xxxxxx
```

where the x's indicate that the message is unintelligible, your terminal is most likely connected to a 3704/3705 line in NCP mode that is defined for a different terminal type. For example, you may have an EBCD terminal on a line defined for Correspondence terminals. Use the @ (at sign) character to determine what kind of terminal you are using. If the @ character is an uppercase 2, your terminal is a Correspondence 2741; otherwise, it is an EBCD 2741.

If a 2741 terminal is connected to VM/370 via a 3704/3705 line in NCP mode, you must press the Return key before the "vm/370 online" message appears at the terminal. If a terminal is connected to VM/370 via a 3704/3705 line in NCP mode, and has the Multiple Terminal Access (MTA) feature, then the "vm/370 online" message does not appear at the terminal and, after approximately 15 seconds, the terminal locks and unlocks. You must perform a special sign-on procedure before continuing with the normal logon procedure.

The sign-on procedures for the terminals supported for use with the 3704/3705 under the control of VM/370 are summarized in the following paragraphs. If you have any further difficulties accessing VM/370 through a 3704/3705, see the VM/370: Terminal User's Guide and the 3704 and 3705 Generation and Utilities Guide for complete descriptions of the procedures summarized here.

FOLLOW THE SPECIAL SIGN-ON PROCEDURES FOR 3704/3705 LINES THAT ARE IN NCP MODE AND ALSO HAVE THE MTA FEATURE

Three sign-on procedures are described: the 2741, 1050, and CPT-TWX for terminals on lines with the MTA feature. Once the sign-on procedure is completed, the message

```
vm/370 online
```

should appear at your terminal. You can then proceed with the normal logon procedure described in the VM/370: Terminal User's Guide.

MTA Sign-on Procedures for IBM 2741 Terminal

1. Dial the telephone number of the MTA line to be used for communicating with the controller.
2. When the keyboard unlocks, enter /".
3. Press the Return key.

If the "vm/370 online" message appears at your terminal, you have signed on successfully and may proceed with the normal logon. If no message appears at your terminal but your terminal unlocks, press the Return key in an attempt to get the "vm/370 online" message. However, if the type element moves back and forth, the sign-on is unsuccessful; you must repeat steps 2 and 3 of the sign-on procedure.

| Note: The 3767 terminal (operating as a 2741) is not suppcrted by NCP
| lines with the MTA feature because of the 300 bits/second line speed.

MTA Sign-on Procedure for IBM 1050 Terminal

1. Dial the telephone number of the MTA line to be used for communicating with the controller.
2. When the Proceed light comes on, enter /".
3. Press the Return key.
4. Enter EOB.

If the "vm/370 online" message appears at your terminal, you have signed on successfully and may proceed with the normal logon. If no message appears at your terminal but your terminal unlocks, press the Return key in an attempt to get the "vm/370 online" message. However, if the type element moves back and forth, the sign-on is unsuccessful; you must repeat steps 2, 3, and 4 of the sign-on procedure.

MTA Sign-on Procedure for CPT-TWX Terminal

1. Dial the telephone number of the MTA line to be used for communicating with the controller.
2. Press the WRU (Where Are You) key within three seconds after the audible data tone begins.

If the typing mechanism does not "jump" within a few seconds, you have signed on successfully and may proceed with the normal logon. If the typing mechanism does "jump," the sign-on is unsuccessful; press the WRU key again or repeat both steps of the sign-on procedure.

LOGGING ON AFTER AN NCP CONTROL PROGRAM HAS ABNORMALLY TERMINATED

If an NCP 3704/3705 control program (with the automatic dump and reload options previously set) abnormally terminates but VM/370 continues to run, VM/370:

1. Disconnects all the 3704/3705 users
2. Dumps the contents of 3704/3705 storage
3. Reloads the 3704/3705 control program
4. Enables the lines again

At this point, each user must log on again. Any user that does not log on with 15 minutes is logged off the system.

STEP 12. APPLYING PTFS TO THE 3704/3705 LOAD LIBRARY

If necessary, it is possible to apply Program Temporary Fixes (PTFs) directly to the 3704/3705 load library. The CMS ZAP program applies the PTF. See the "ZAP Service Program and How to Use It" section of Part 6.

After you have generated a 3704/3705 control program, loaded it, and logged on, you may want to test the 3704/3705 control program. Several CP commands are provided to control the operation, check the status, and dump the contents of the 3704/3705. The NETWORK command loads and dumps any 3704/3705 control program. It also controls the operation of NCP and PEP 3704/3705 control programs, while the existing CP commands (ENABLE, DISABLE, QUERY, DISPLAY, VARY, and HALT) provide similar support for EP 3704/3705 control programs. The NCPDUMP command formats and prints a dump of 3704/3705 storage. Use these commands to test the 3704/3705 control program.

The NETWORK, ENABLE, DISABLE, QUERY, DISPLAY, VARY, and HALT commands are described in the VM/370: Operator's Guide and the VM/370: Command Language Guide for General Users.

NCPDUMP SERVICE PROGRAM AND HOW TO USE IT

NCPDUMP is a CMS command. It processes CP spool reader files created by 3705 dumping operations, that is, dump files that are produced as a result of the CP NETWORK command specified with the DUMP operand and either automatic or immediate mode.

The NCPDUMP file processing operation can include:

- Erasing a specific CMS NCPDUMP file after printing it
- Formatting the dump
- Printing the dump
- Assigning an identifier to the CMS NCPDUMP file
- Creating the CMS NCPDUMP file from the spool file

Although NCPDUMP is a CMS command, its use is restricted to the user identified by the SYSDUMP operand of the SYSOPER macro in DMKSYS during VM/370 system generation. The operation of NCPDUMP is similar to VMFDUMP operations. A general description of the NCPDUMP operation follows the command description.

The NCPDUMP command has the following format:

```
NCPDUMP | [DUMPxx] [([ERASE][NOFORM][MNEMONIC][)]]
```

where:

DUMPxx is the filename of a CMS file containing a 3704/3705 Communications Controller program dump. This dump was created by a previously invoked NCPDUMP command with the ERASE operand not specified.

ERASE erases the current CP DUMP file or a specified DUMPxx (filename), saved CMS file.

NOFORM specifies that a formatted control block is not desired.

MNEMONIC includes 3705 Assembler mnemonic operation codes in the printed output.

The NETWORK command invoked with the DUMPxx operand, as stated previously, produces CP files that contain the contents of a designated 3704/3705 Communications Controller unit buffer. These CP files reside as a spooled reader input assigned to a system-designated user. The CMS NCPDUMP command invoked by this user formats (if requested) and prints the contents of these files.

The NCPDUMP program creates a CMS file with a filename DUMPxx and a filetype of NCPDUMP, and erases the original spooled NETWORK initated dump reader file. The created CMS file is erased if you specify ERASE, otherwise it is kept.

A maximum of ten dumped spooled files can be processed and saved, and later recalled, if necessary, by the system assignment of an xx identifier suffix to the CMS DUMPxx filename. The "xx" is a decimal number from 00 to 09, depending on any existing files of a similar name. For example, if the files DUMP00 NCPDUMP and DUMP01 NCPDUMP already exist, the new file would be called DUMP02 NCPDUMP. The file thus created is retained for later use unless the ERASE option is specified, in which case the file is erased immediately after the dump is printed.

Part 5: The Standalone Spool Remote Program

Part 5 describes the procedure for installing the standalone spool remote program (DMKSRP) for the IBM 2780. It contains an introduction to the 2780 and information about:

- Installing the DMKSRP program
- Testing 2780 operation

The standalone spool remote program (DMKSRP) transfers data between a VM/370 spool file and a 2780. Printer files are sent in nontransparent mode. To increase throughput, VM/370 removes all trailing blanks from each print line before sending them. Printed output may be stopped at any time to send a control card to the system (but punched output may be stopped for this purpose only if the 2780 is in Allow Punch Interrupt [API] mode). See the VM/370 Operator's Guide for a description of API mode.

IBM 2780 DATA TRANSMISSION TERMINAL CHARACTERISTICS

The IBM 2780 Data Transmission Terminal (Model 2) can be used as a remote spooling work station under VM/370. It provides medium-speed input and output capabilities over common carrier communications facilities. The 2780 Model 2 is used for card input, printed output, and punched output.

The "Configurations" section of Part 1 lists the 2780 features that the VM/370 standalone spool remote program requires.

When you initially install the DMKSRP program, or make changes to it, you must code and assemble the SRP2780 macro. The text file produced by the assembly is placed on your A-disk.

To install the DMKSRP program, you must:

- Format your A-disk.
- Code the SRP2780 macro.
- Assemble the DMKSRP program.

The following procedure describes the installation process.

STEP 1. FORMAT YOUR A-DISK

Log on VM/370, IPL a CMS virtual machine and format your A-disk. Use the CMS FORMAT command to format your A-disk.

STEP 2. CREATE A DMKSRP ASSEMBLE FILE

Use the CMS Editor to create a file called DMKSRP ASSEMBLE. The file must contain the SRP2780 macro and an END statement. The END statement must specify DMKSRP80 as the entry point.

The format of the SRP2780 macro is:

Name	Operation	Operands		
label	SRP2780	IPLUNIT=cuu, IPLUNIT=00C,	PUNUNIT=cuu, PUNUNIT=00D,	PRTUNIT=cuu, PRTUNIT=00E,
		REMUNIT=cuu, REMUNIT=018,	PRTRDR=cuu, PRTRDR=010,	PUNRDR=cuu, PUNRDR=011,
		CONUNIT=cuu CONUNIT=009		

where:

label
is any assembler language label.

IPLUNIT=cuu
is the virtual address (cuu) of the device that loads the DMKSRP program. The default value is X'00C'.

PUNUNIT=cuu
is the virtual device address (cuu) of the virtual unit record device that DMKSRP uses for punching to a real card punch and for sending remote card files to other users. The real card punch can be set at the central computing location or at a remote location. The default value is X'00D'.

PRTUNIT=cuu
is the virtual address (cuu) of the virtual unit record device that the DMKSRP program uses for printing to the real printer at the central computing location. The default value is X'00E'.

REMUNIT=cuu
is the virtual address (cuu) of the binary synchronous communications line attached to the 2780 work station. The default value is X'018'.

PRTRDR=cuu
is the virtual address (cuu) of the virtual reader containing print files for the 2780 work station. The default value is X'010'.

PUNRDR=cuu
is the virtual address (cuu) of the virtual reader containing punch files for the 2780 work station. The default value is X'011'.

CONUNIT=cuu
is the virtual address (cuu) of the console used to communicate with the DMKSRP program. The default value is X'009'.

Note: You can specify the same virtual address for PRTRDR and PUNRDR.

An example of a DMKSRP ASSEMBLE file is:

```
SRP      TITLE   'DMKSRP VERSION v, LEVEL 1...'
RJE2780  SRP2780 IPLUNIT=00C,PUNUNIT=00D,PRTUNIT=00E,          X
                 REMUNIT=018,PRTRDR=010,                       X
                 PUNRDR=011,CONUNIT=009
         END     DMKSRP80
```

STEP 3. USE VMFASM TO ASSEMBLE DMKSRP

Issue the VMFASM command to assemble DMKSRP and apply any updates that exist. For example,

```
VMFASM DMKSRP DMKR20
```

VMFASM places the text deck it creates on your A-disk. VMFASM is described in "Part 6: Updating VM/370."

STEP 4. LOG ON THE 2780 VIRTUAL MACHINE

If you are installing the 2780 spool remote program, your VM/370 directory should have an entry for a virtual machine to execute the 2780. The VM/370 Directory program control statements that are distributed with the starter system have an entry for a 2780 virtual machine. Refer to Part 2 to see that entry. The following VM/370 directory is representative of a 2780 virtual machine.

```
USER  REM2780 REMOTE 320K
ACCOUNT number
CONSOLE 009 3210
SPOOL 00E 1403 A
SPOOL 00D PUNCH A
SPOOL 00C 2540 READER A
SPOOL 010 2540 READER A
SPOOL 011 2540 READER A
OPTION REALTIMER
MDISK 191 3330 003 001 CPV1L0 W
LINK CPSYS 190 190 R
LINK CPSYS 194 194 R
DEDICATE 018 018
```

The DEDICATE control statement defines a transmission control unit with a binary synchronous communication (BSC) line that connects the 2780 to this virtual machine (REM2780). If you do not include the DEDICATE statement in the directory entry, the user of the 2780 virtual machine must explicitly request that the real BSC line be attached to his 2780 virtual machine when he logs on. The REALTIMER option is required for the DMKSRP program error recovery procedures.

STEP 5. PUNCH THE DMKSRP TEXT FILE

Make the text deck created in Step 3 available to your virtual machine as an input spool file. Issue the following CMS commands:

```
spool pch to * cont
punch 3card loader (noheader)
punch dmksrp text (noheader)
spool punch nocont
close pch
```

STEP 6. LOAD THE DMKSRP PROGRAM

You can then IPL the DMKSRP program by entering CP mode and issuing the IPL command, as follows:

```
ipl 00c
```

00C is the address of the virtual card reader that contains the text file punched in Step 5.

STEP 7. ENABLE THE LINE

When the 2780 virtual machine is initialized, the message ENABLING is displayed at the terminal. At this time, you can disconnect your terminal and make the remote terminal available to another user.

If your 2780 is on a point-to-point line, the connection between the 2780 and the central computer is established when the enable is performed. If your 2780 is on a switched line, you must determine which work station is to be called, make sure the data set at that work station is in auto answer mode, and then call the station to establish the line connection.

Note: You can perform Steps 5, 6, and 7 by invoking a CMS EXEC procedure, such as:

```
&CONTROL OFF
CP CLOSE PCH
CP SPOOL PCH TO * CONT CLASS Q
PUNCH 3CARD LOADER (NOHEADER)
PUNCH DMKSRP TEXT (NOHEADER)
CP SPOOL PUNCH NOCONT
CP CLOSE PCH
CP SPOOL PCH OFF CLASS A
CP SPOOL 00C CLASS Q
CP DISCONN
CP IPL 00C CLEAR
```

If you use the preceding EXEC procedure, you should change 00C to the address you specified for IPLUNIT on the SRP2780 macro. Use a spooling class in the third and ninth lines that allows the DMKSRP program to be loaded whether or not there are other spool files in the virtual reader. The class letter Q has no special significance in this example; you should use a spooling class that is not normally used by your installation. (The file 3CARD LOADER is found on the VM/370 system update disk, along with the DMKxxx ASSEMBLE files.)

If this EXEC procedure is used, the ENABLING message does not appear at your terminal because your virtual machine is disconnected before DMKSRP is loaded.

CONTROLLING 2780 OPERATION

Once a connection is established between the 2780 and the VM/370 system, control is retained or changed through a set of DMKSRP control cards. The 2780 operator creates these cards and puts them in the 2780 cards reader. These control cards are described in the VM/370: Operator's Guide.

TESTING 2780 OPERATION

You can verify that the DMKSRP program is operating correctly by performing the following tests from another virtual machine (not the 2780's virtual machine) that has access to CMS disk files:

```
SPOOL D TO REM2780
SPOOL E TO REM2780
```

where REM2780 is the userid of the virtual machine executing the 2780 spool remote program. Then issue both of the following commands:

```
PRINT fname ftype
PUNCH fname ftype
```

At the 2780, have a CONTINUE control card and an ID control card prepared, specifying on the ID card the VM/370 userid of the testing virtual machine. Place the CONTINUE card in the 2780 reader hopper. Ready the printer, press the reader end-of-file pushbutton, and press the reader START pushbutton twice. Printing should begin at the 2780.

When the TERM ADR light comes on and the alarm sounds, rotate the MODE switch to reset the 2780. Place a CONTINUE control card, followed

by blank cards, in the hopper and ready the 2780. Punching shculd begin.

When the 2780 activity stops, remove the remaining blank cards from the reader hopper and place the ID card, followed by the deck that was punched at the 2780, in the reader hopper. Be sure to remove the blank cards preceding and following the data cards that were punched. Once again, ready the printer, press the reader end-of-file pushbutton and press the reader START pushbutton twice. The card deck should be read at the 2780 and transferred to the testing virtual machine, which should then issue the command:

```
READCARD fname ftype
```

to read the card deck and make it a CMS file. Compare the original file and the file which was read back to verify the correct operaticn of the 2780 spool remote program.

Under normal operation, the 2780 printer is ready, with the rotary switch on the reader/punch set on transmit transparency. Then printed spooled output can be received, and control cards can be sent through the reader. If the terminal address (TERM ADR) light comes on and the audible alarm sounds, the DMKSRP program is attempting to punch spooled output. Perform a nonprocess runout to clear the punch. Rotate the MODE switch to reset the 2780. Place a CONTINUE control card, followed by blank cards, in the hopper. Press the START key on the reader/punch. The 2780 should begin to punch the spooled output.

Part 6: Updating VM/370

Part 6 tells you how to apply Program Temporary Fixes (PTFs) and updates to an installed VM/370 system. It contains information about the following:

- Supporting a VM/370 System
- Updating Modules Using the VMFASM EXEC Procedure
- Using the VMFMAC EXEC Procedure to Update Macro Libraries
- Using VMFLOAD to Generate a New Nucleus
- The Loader
- Using the GENERATE EXEC Procedure to Generate a New CP or CMS Nucleus
- Using the VMFBLD EXEC Procedure to Build a New Nucleus
- Using the CMSGEND EXEC Procedure to Generate a CMS Module
- Using the ASMGEND EXEC Procedure to Generate the Assembler
- Recommended Procedures for Updating VM/370

The multiple virtual machine environment created by VM/370 permits support of both hardware and software to be done concurrently with other installation work.

Virtual machines can be used to:

- Generate and test new systems
- Apply and test PTFs
- Run hardware diagnostics
- Retrieve and examine VM/370 ABEND dumps and error recordings
- Examine portions of real VM/370 storage
- Trace the execution of a system in a virtual machine

Before installing VM/370, you should develop an account support plan with the IBM FE program systems representative. Appropriately configured virtual machine entries should be included in the VM/370 directory for the service representative. Two virtual machines, with userids CE and MAINT, are defined for these representatives in the VM/370 directory distributed with the starter system.

VM/370 UPDATE PROCEDURES

Using the VM/370 update facility, you can update files with several levels of updates and/or any number of Program Temporary Fixes (PTFs). Procedures are supplied for assembling the updated source code to produce a uniquely identifiable text file. The file has a unique filename and records that identify the origin of the updates, macro libraries, and source statements.

Procedures are provided for generating load files from various object modules, and for generating MACLIB files from various COPY and MACRO files.

The update structure involves a file naming convention for update and text files, a set of programs to support the processing, and a set of EXEC procedures to process the files.

The update procedures and programs supplied with VM/370 are:

VMFASM	Incorporates PTFs and/or updates and creates a new text file
VMFLOAD	Generates a new CP, CMS, or RSCS module
CMSGEND	Generates a new CMS module
GENERATE	Generates a new VM/370 system
GENERATE IPIDECK	Generates a new standalone version of a service program on disk
GENERATE SRVCPGM	Punches the service programs on cards
VMFMAC	Generates a new CMS or CP macro library
CMS UPDATE Command	Updates modules

All modules prefaced by the letters DMK are CP modules. There are two kinds of CP modules: those that are part of the CP nucleus (these modules are contained in the CPLOAD EXEC file) and those that are not part of the CP nucleus (service programs that execute either standalone or under CMS). The programs that execute standalone are DMKDDR, DMKDIR, and DMKFMT. If you apply a PTF to these modules and create a new text deck, use the GENERATE EXEC to create a new standalone file.

The service programs that execute under CMS are:

- DASD Dump Restore Program (module DMKDDR)
- Directory program (module DMKDIR)
- VMFDUMP, the virtual dump program (module DMKEDM)
- NCPDUMP, the 3704/3705 dump program (module DMKRND).

If you apply updates to these modules, use the CMSGEND EXEC to create a new CMS module. The module name to specify is DDR, DIRECT, NCPDUMP, or VMFDUMP, respectively. CMS cannot execute the Format/Allocate program (module DMKFMT) and the IBCDASDI Virtual Disk Initialization program (module IBCDASDI). If you apply a PTF to any DMK module other than these six service programs, you must reload CP (using the VMFLOAD command).

UPDATING A MODULE

The following discussions assume that areas containing the source code for CP and CMS are added to the appropriate virtual machine configurations. Source code for the CMS system is included on the CMS2 tape; source code for CP is included on the CP2 tape. These tapes are distributed by the Program Information Department (PID).

VM/370 has update procedures to incorporate changes (additions, deletions, or corrections) into an existing module, macro, or copy file. For example, if you apply updates to the DMKVAT module, the VMFASM update procedure:

- Locates the DMKVAT source file. It is the unmodified Assembler Language source code that is distributed with the VM/370 system.
- Locates the update control file for DMKVAT. The control file name is specified on the VMFASM command. The control file can have any filename, but must have a filetype of CNTRL. It contains records indicating how to apply the updates.
- Applies the updates to the specified source (in this case, DMKVAT) and gives the updated source a temporary name by concatenating a $ (dollar sign) to the first seven characters of the filename (in this case, the temporary filename is $DMKVAT).
- Puts macro library names specified in the control file into the proper Assembler library list. (The macro libraries required at assembly time are specified in a MACS record in the control file.)
- Assembles the updated source file ($DMKVAT) and creates an updated object deck. The object deck filetype is derived from information found in the control file. The filename of the updated object file is the same as that of the original source, DMKVAT.

As the VMFASM update procedure progresses from one step to the next, informational or error messages are displayed. To more fully understand

how the update procedures operate, you need to know what a control file contains, and how it is handled. The following discussion provides this information.

CONTROL FILES

Control files are used by the CMS UPDATE command and by the VMFASM EXEC procedure. You may have one or more control files to specify various combinations of updates and macro libraries to be used at different times. A control file contains the following types of records:

- The MACS Record. This record contains the names of macro libraries to be used at assembly time. A MACS record is required for a control file used by the CMS UPDATE command; it is optional for a control file used by the VMFLOAD EXEC procedure. A control file must not contain more than one MACS record. If a MACS record is included, it must precede any other records except comments. Up to eight libraries may be specified in the MACS record (if space permits). A MACS record has the following format:

  ```
  uplevel MACS lib1 lib2 lib3...
  ```

  The uplevel field is usually TEXT; it is not used to generate the filetype of the updated file.

- Update identification records. These records identify updates that are to be applied to a particular source file, if such a file exists. An update identification record has the following format:

  ```
  uplevel upid
  ```

  where uplevel is an update level identifier that generates a filetype for the new file (the new filetype uniquely identifies the updated source file). The field, upid, is the update identification; it can be from one to four characters long. This update identification is concatenated with the prefix UPDT to identify the filetype of the direct update to be applied. The filename must be the same as the name of the file to be updated. For example, if an update identification record for DMKVAT is:

  ```
  TEXT NEW1
  ```

  the update file is called DMKVAT UPDTNEW1. The update level identifer is TEXT.

- AUX file identification records. These records contain the names of auxiliary files (AUX files), which in turn contain a list of filetypes of update files to be applied to a particular source file. An auxiliary file identification record has the following format:

  ```
  uplevel AUXnnnn
  ```

  where uplevel is an update level identifier that generates the filetype of the updated source file. The string, nnnn, is an identification string that can be from one to four characters long. This identification string, with the prefix AUX, is the filetype of the auxiliary file that contains the list of updates to be applied. The filename of the auxiliary file is the same as the filename of the source file to be updated. For example, if an AUX file identification record that contains a list of updates for DMKVAT is:

```
TEXT AUXnnnn
```

then the auxiliary file called DMKVAT AUXnnnn contains the filetypes of the update files that are to be applied. These update files have the same filename as the source file to be updated.

- Comments. An asterisk (*) in the first column of the record identifies a comment record.

Note: Control file records in the format that was used under VM/370 Release 1 are still accepted under Release 2. For example, an AUX file identification record in the format

```
TEXT nnnn AUX
```

is still accepted by the update procedures.

A control file can have any number of update identification records, AUX file identification records, and comments, but can have only one MACS record. The control file can have any filename. Note, however, that VM/370 updates from IBM normally use the following special ccntrol files:

- A control file for CP source, copy, and macro updates is called DMKR20 CNTRL.
- A control file for CMS source updates is called DMSR20 CNTRL.
- A control file for CMS macro and copy updates is called DMSM20 CNTRL.
- A control file for assembling the NCPDUMP source is called NCPR20 CNTRL.
- A control file for assembling RSCS source, copy, and macro updates is called DMTR20 CNTRL.

The DMKR20 CNTRL file contains the following records:

```
TEXT  MACS DMKMAC CMSLIB OSMACRO
TEXT  AUXR20
```

The DMSR20 CNTRL file contains the following records:

```
TEXT  MACS CMSLIB OSMACRO
TEXT  AUXR20
```

The DMSM20 CNTRL file contains the following record:

```
COPY  AUXM20
```

The NCPR20 CNTRL file contains the following records:

```
TEXT  MACS OSMACRO DMKMAC CMSLIB
TEXT  AUXR20
```

The DMTR20 CNTRL file contains the following records:

```
TEXT  MACS DMTLOC DMTMAC
TEXT  AUXR20
```

APPLYING PTFS TO VM/370

PTF updates are distributed in card or magnetic tape form, or as APAR answers typed on coding forms. In any case, a CMS file (with the correct filename and filetype) must be created on a disk to contain the update. The disk must belong to the user (userid MAINT) who is responsible for updating VM/370. The disk may be the CP or CMS source disk, but it is usually a separate disk.

A suggested virtual machine configuration for updating a 2314 system is:

```
USER MAINT CPCMS 512K 16M BCEG
  ACCOUNT (installation defined)
    OPTICN ECMODE REALTIMER
    CONSOLE   009   3215
    SPOOL     00C   2540   READER A
    SPOOL     00D   2540   PUNCH  A
    SPCOL     00E   1403   A
    MDISK   190 2314 035 110 CPV2L0 MR READ
    MDISK   191 2314 019 010 CPV2L0 MR READ
    MDISK   194 2314 145 058 CPV2L0 MR READ
    MDISK   199 2314 034 001 CPV2L0 WR READ
    MDISK   193 2314 001 050 USERD1 MR READ
    MDISK   294 2314 051 050 USERD1 MR READ
    MDISK   393 2314 001 110 USERD2 MR READ
    MDISK   394 2314 001 110 USERD3 MR READ
    MDISK   390 2314 101 003 USERD1 MW READ
    MDISK   cuu 2314 000 203 yyyyyy MW
```

where cuu and yyyyyy are the address and label of your system residence volume defined in your DMKSYS module.

A suggested virtual machine configuration for updating a 3330 system is:

```
USER MAINT CPCMS 512K 16M BCEG
  ACCOUNT (installation defined)
    OPTION ECMODE REALTIMER
    CONSCLE   009   3215
    SPOOL     00C   2540   READER A
    SPCOL     00D   2540   PUNCH  A
    SPOOL     00E   1403   A
    MDISK   190 3330 030 076 CPV2L0 MR READ
    MDISK   191 3330 016 007 CPV2L0 MR READ
    MDISK   194 3330 106 044 CPV2L0 MR READ
    MDISK   199 3330 029 001 CPV2L0 WR READ
    MDISK   193 3330 001 030 USERD1 MR READ
    MDISK   294 3330 031 030 USERD1 MR READ
    MDISK   393 3330 061 060 USERD1 MR READ
    MDISK   394 3330 121 060 USERD1 MR READ
    MDISK   390 3330 181 002 USERD1 MW READ
    MDISK   cuu 3330 000 403 yyyyyy MW
```

where cuu and yyyyyy are the address and label of your system residence volume defined in your DMKSYS module.

A suggested virtual machine configuration for updating a 3340 system is:

```
USER MAINT CPCMS 512K 16M BCEG
  ACCOUNT (installation defined)
    OPTION ECMODE REALTIMER
    CONSCLE   009   3215
    SPOOL     00C   2540   READER A
    SPCOL     00D   2540   PUNCH  A
    SPOOL     00E   1403   A
    MDISK 190 3340 030 076 CPV2L0 MR READ
    MDISK 191 3340 016 007 CPV2L0 MR READ
    MDISK 194 3340 106 044 CPV2L0 MR READ
    MDISK 199 3340 029 001 CPV2L0 WR READ
    MDISK 193 3340 001 030 USERD1 MR READ
    MDISK 294 3340 031 030 USERD1 MR READ
    MDISK 393 3340 061 060 USERD1 MR READ
    MDISK 394 3340 121 060 USERD1 MR READ
    MDISK 390 3340 181 002 USERD1 MW READ
    MDISK cuu 3340 000 348 yyyyyy MW
```

where cuu and yyyyyy are the address and label of your system residence volume defined in your DMKSYS module.

The entries in the preceding VM/370 directory, with the exception of the 192, 294, 393, 394, and 390 virtual disks, are included in the 2314, 3330, and 3340 VM/370 directories supplied with the starter system, and should be included in your VM/370 directory, as they are used by IBM for support.

The contents of the preceding virtual disks are:

Disk	Contents
190	Current CMS system disk
191	Work area
194	CP text retention
199	The 191 minidisk (work area)
193	CMS PTFs, updates, and updated text decks (object modules)
294	CP PTFs, updates, and updated text decks (object modules)
393	CMS source and macros
394	CP source, macros, and copy files
390	CMS test nucleus area
cuu	CP system residence device, or a replica of it, for test purposes

These virtual disks are shown in Figure 27.

You should apply all distributed updates. Once you create the appropriate files, you should access the disks containing the CP, CMS, or RSCS source files and update procedures, and apply the updates.

To apply the IBM distributed updates to an existing source file, use the VMFASM EXEC procedure. To apply the IBM distributed updates to a copy or macro file, use the VMFMAC EXEC procedure.

If you update a copy or macro file, you should use the VMFASM EXEC procedure to reassemble the module(s) that contain that copy or macro file.

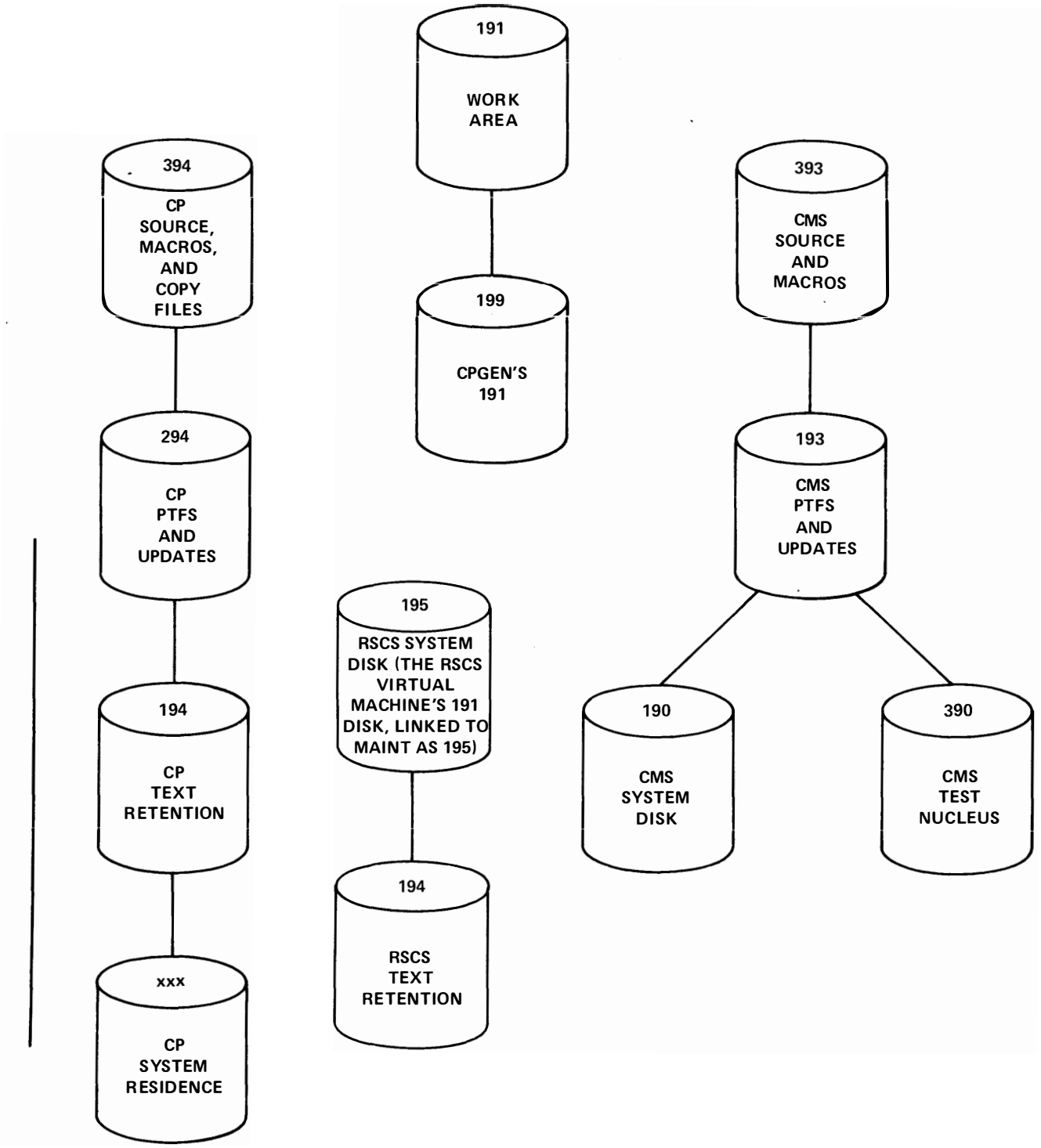
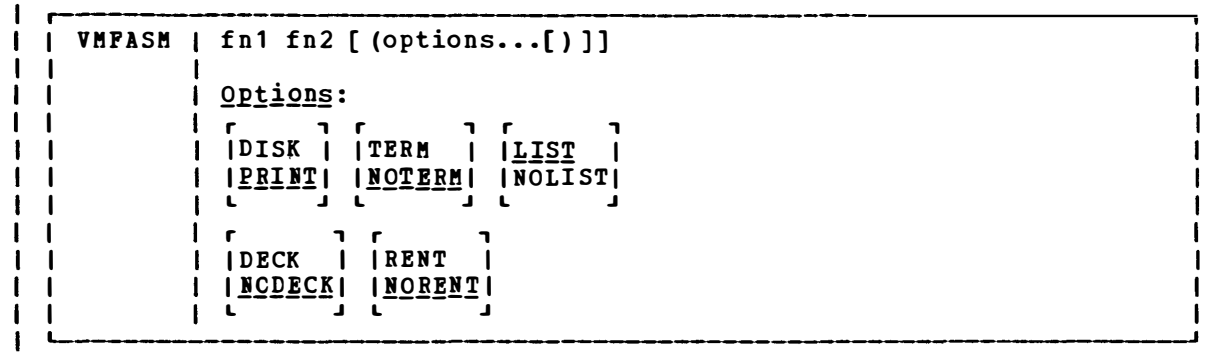


Figure 27. System Support Plan

Updating Modules Using the VMFASM EXEC Procedure

Use the VMFASM EXEC procedure to update a specified source file according to entries in a control file, and to assemble the updated source file. VMFASM invokes the CMS UPDATE command. The format of the VMFASM command is:



where:

fn1 is the filename of the source file to be updated.

fn2 is the filename of the control file. The control file must have a filetype of CNTRL.

Options:

DISK places the LISTING file on a virtual disk.

PRINT writes the LISTING file to the printer.

TERM writes the diagnostic information on the SYSTERM data set. 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 writes an object module on the device specified on the FILEDEF statement for PUNCH.

NODECK suppresses the DECK option.

RENT checks the program for a possible violation of program reenterability. Code that makes the program nonreenterable is identified by an error message.

NORENT suppresses the RENT option.

USING VMFASM TO APPLY IBM-SUPPLIED UPDATES

The control file contains records that identify the updates to be applied and the macro libraries, if any, needed to assemble the source program. The updates are applied starting with the last update file in the control file and in sequence up to the first update file (or the update file immediately following the MACS record). Updates identified by auxiliary files are applied starting with the last update in the auxiliary file and proceeding in sequence up to the first.

For example, a control file named UPDATE1 CNTRL contains the following two records:

```
TEXT   MACS   DMKMAC   CMSLIB   OSMACRO
IBM1   AUX2000
```

An Assembler Language source file is named DMKVAT ASSEMBLE.

An auxiliary file named DMKVAT AUX2000 contains a list of filetypes (NEW2 and NEW1) with NEW2 the first entry and NEW1 the last.

The two update files are named DMKVAT NEW1 and DMKVAT NEW2. These are the files identified by the auxiliary file, DMKVAT AUX2000. Assume these files contain IBM-supplied updates to DMKVAT, such as inserted, deleted, or replaced source statements and the appropriate control statements. The update control statements are described in the VM/370: Command Language Guide for General Users with the CMS UPDATE command.

To update DMKVAT, you enter the command:

```
VMFASM DMKVAT UPDATE1
```

VMFASM does the following:

- VMFASM locates the DMKVAT ASSEMBLE and UPDATE1 CNTRL files.
- The UPDATE1 CNTRL file is processed from the bottom up. The first entry found is IBM1 AUX2000.
- VMFASM tries to locate the file named DMKVAT AUX2000 by searching all accessed disks.
- When VMFASM locates the DMKVAT AUX2000 file, it processes it from the bottom up. The first entry found is NEW1.
- VMFASM tries to locate the the update file named DMKVAT NEW1. When it finds DMKVAT NEW1, VMFASM applies the updates that are in DMKVAT NEW1 to the DMKVAT ASSEMBLE file, and creates a new file called $DMKVAT ASSEMBLE.
- Next, VMFASM processes the NEW2 entry in the DMKVAT AUX2000 file. When VMFASM locates the update file DMKVAT NEW2, it applies the updates to the updated ASSEMBLE file ($DMKVAT).
- Because there are no more filetypes listed in the DMKVAT AUX2000 file, VMFASM reads the next control record in the UPDATE1 CNTRL file. In this case, it is the MACS record.
- After entering the macro library names that are on the MACS record into the appropriate Assembler library lists, VMFASM assembles the updated ASSEMBLE file ($DMKVAT).
- The UPDATE command then stacks in the console read buffer the uplevel (update level identifier) associated with the last update applied. If

there were no updates, it stacks the uplevel associated with the MACS control record and the names of the macro libraries specified in the MACS record. VMFASM then reads the stacked lines and concatenates the uplevel (if it is not TEXT) to the characters TXT to form the filetype of the assembled updated source.

An update level identifier of TEXT causes special handling in the VMFASM EXEC procedure, whether or not an update is used with it. A name of TEXT is used as the object module filetype without level identification concatenation. Thus, TEXT becomes the filetype.

VMFASM places the macro library names that were specified on the MACS record in the Assemble library list (via the CMS GLOBAL command) so that those libraries can be used when the updated source file is assembled.

In this example, the last (and only) update applied was identified as IBM1 AUX2000. The file identification for the updated source is DMKVAT TXTIBM1. The updated source is assembled using the macro libraries DMKMAC, CMSLIB, and OSMACRO.

You may want, on occasion, to have entries in a control file that specify an update level identifier but no update. A record of the following format, for example, is allowed:

```
NAME5
```

because the control file is used for loading object modules (text decks) as well as for updating input files.

If updates are not found, a message is issued and processing continues, if possible.

USING VMFASM TO APPLY YOUR OWN UPDATES

If you wish to apply your own update to VM/370 (for example, if you wish to expand the accounting routines), you follow the same procedure described for applying IBM-supplied updates.

You create the update file. You can name the update file in either of two ways. If you are going to identify the update file directly in the control file, use the form:

```
DMKACO UPDTupid
```

where DMKACO is the filename of the accounting module you wish to expand, and upid is the identification for the filetype. For example, you might call your update file:

```
DMKACO UPDTFIX1
```

The second way to identify your update is via an auxiliary file. If you use an auxiliary file, you use the following form to name your update file:

```
DMKACO ft
```

where DMKACO is the filename of the accounting routine you wish to expand and ft is any filetype.

For example, you could have two update files called:

```
DMKACO NEW1
DMKACO NEW2
```

When you decide to use an auxiliary control file, it must have a name in the form

```
DMKACO AUXnnnn
```

For example, assume you have an auxiliary control file called:

```
DMKACO AUX1111
```

This AUX file has the following entries (the filetypes of your update files):

```
NEW2
NEW1
```

Next, you must create a control file to identify all IBM-supplied updates to the module you are changing and your own updates. You must apply the IBM-supplied updates first. Assume there are IBM-supplied updates in an auxiliary file called:

```
DMKACO AUXR20
```

and that your own updates are those used as examples in the preceding paragraphs. Then, you need your own control file, identified as:

```
fn CNTRL
```

It can have any filename, but its filetype must be CNTRL. For this example, the control file is called:

```
LOC CNTRL
```

and it has the following records:

```
TEXT MACS DMKMAC
LOCAL FIX1
SPEC AUX1111
IBM1 AUXR20
```

To apply the updates to DMKACO, issue the command:

```
VMFASM DMKVAT LOC
```

The VMFASM procedure handles the update as follows; it:

- Locates the source file, DMKACO.
- Locates the control file, LOC CNTRL.
- Reads the control file, last line first (IBM1 AUXR20).
- Locates the IBM-supplied auxiliary file, DMKACO AUXR20.
- Reads the DMKACO AUXR20 auxiliary file from bottom to top and applies the IBM-supplied updates to DMKACO, naming the updated source $DMKACO ASSEMBLE.
- Reads the next entry in the control file (SPEC AUX1111).

- Locates your own auxiliary file, DMKACO AUX1111.
- Reads the last entry (NEW1), locates the update file DMKACO NEW1, and applies the update to the updated source $DMKACO.
- Reads the next entry in your auxiliary file (NEW2), locates the corresponding update file DMKACO NEW2, and applies it to the updated source $DMKACO. Processing for your auxiliary file is now complete.
- Reads the next entry in the control file (LOCAL FIX1).
- Locates the directly-identified update file, DMKACO UPDTFIX1.
- Applies the DMKACO UPDTFIX1 updates to the updated source ($DMKACO).
- Reads the next control record, the MACS record.
- Issues the GLOBAL command for the macro libraries identified on the MACS record (DMKMAC MACLIB).
- Assembles the updated source file, $DMKACO ASSEMBLE.
- Names the object module DMKACO TXTLOCAL. The filetype is derived by concatenating the prefix (TXT) and the uplevel of the last update applied (LCCAL).

If you create a new object module for a VM/370 module, you must also reconstruct the CP, CMS, or RSCS nucleus using the VMFLOAD service program. Or, if the file is a part of a CMS command module, use the CMSGEND procedure to generate a new module utilizing the new object code. See Appendix D to determine whether the CMS nucleus or some other MODULE files must be generated again because of your update.

When you use CMSGEND to create a new module, you must change the filetype of the object deck to TEXT (if it is not already TEXT) before issuing the CMSGEND command. After the CMSGEND processing is complete, you can change the filetype of the object deck back to what it was before.

Note that CMSGEND renames the existing module to "fname MODOLD" before creating the updated module. This ensures that any users currently using the CMS system do not have their processing interrupted by the updating of modules, because the SSTAT (system status table) of the loaded system is still pointing to the area on the system disk occupied by the renamed module. When the system is reloaded, the SSTAT points to the updated module, and the old module can be erased.

OTHER FILES PRODUCED BY VMFASM

VMFASM invokes the UPDATE command, which produces two output files that indicate which updates were applied. The file

fn UPDATES

lists the names of update files that were applied to the source file (fn) and the file

fn UPDLCG

lists the actual updates that were applied to the source file (fn) and error messages. Both of these files are included in the LISTING file, and precede the program source statements. Also, the file (fn UPDLOG) precedes the object code in the text file.

Using the VMFMAC EXEC Procedure to Update Macro Libraries

Use the VMFMAC EXEC procedure to update macro libraries. It invokes the CMS UPDATE command to update specified copy 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.

The format of the VMFMAC command is:

```
| VMFMAC | libname control                                                |
```

where:

libname is the filename of an EXEC file that contains the filenames of macro or copy files to be updated. The entries in the libname file are in the following format:

```
&1 &2 fname1
&1 &2 fname2
      .
      .
      .
```

where fname1, fname2, and so on, are filenames of macro or copy files to be updated and included in the macro library (MACLIB).

control is the filename of a control file to be used to apply the updates. The filename is usually DMSnnn, DMKnnn, or DMTnnn, and the filetype must be CNTRL.

VMFMAC finds the EXEC file specified by libname, and invokes the CMS UPDATE command to apply updates, in the order indicated by the control file, to the macro or copy files listed in the libname EXEC file. VMFMAC then builds a new macro library containing the updated files.

VMFMAC produces two files: libname MACLIB and libname COPY. The file, libname MACLIB, is the new macro library. The file, libname COPY, contains a list of the updates applied to each macro. The update log and a copy of the updated macro is printed.

If a file named libname MACLIB already exists, VMFMAC renames it libname OLDMAC. However, if libname OLDMAC already exists, processing is terminated.

If a file named libname COPY already exists, VMFMAC renames it libname OLDCOPY. However, if libname OLDCOPY already exists, processing is terminated.

Copy files should contain '*COPY copyname' as the first record. If this record is not included, the copyname for the member becomes the filename used in the MACLIB command that added the member. This allows copyname to be put in the MACLIB directory regardless of the filename of the copy file.

If any errors are encountered during the update process, processing terminates.

Using VMFLOAD to Generate a New Nucleus

Use the VMFLCAD program to generate a new CP, CMS, or RSCS nucleus. The VMFLOAD program uses two files, a loadlist EXEC file and a control file, to produce a punch file that has several object modules. VMFLOAD requires a virtual machine with 320K.

The format of the VMFLOAD command is:

```
VMFLOAD | loadlist control
```

where:

loadlist is the filename of an EXEC file that contains the names of object modules in the order in which they are to reside in the complete load file for the nucleus. For example:

```
&CONTROL OFF
&1 &2 &3 filename [filetype]
&1 &2 &3 filename [filetype]
 .
 .
 .
```

The filename is the name of an object module to be punched. The object modules are punched in the order specified. If a filetype is specified, VMFLOAD searches for that specific file, and, if it finds it, punches it without a header card.

If the filetype is not specified in the loadlist, VMFLOAD uses the control file to determine which object module is the highest level object module available. VMFLOAD searches the control file from top to bottom. When it finds the appropriate object module, VMFLOAD punches it.

control is the filename of the control file. This is usually the same control file used to apply updates to modules via the VMFASM or UPDATE commands. This file identifies the highest level object module available, if the filetype is not specified in the loadlist.

The VMFLOAD program displays a confirmation message or error message upon completion. Before the loadlist procedure is invoked, a SPOOL PCH CONT command is executed to ensure that the punched files appear as one deck. The commands SPOOL PCH CONT and CLOSE PCH are executed upon completion. You may wish to specify SPOOL PCH TO userid to transfer the punched output as a file to your own (or another) virtual machine as a reader file. If you want to perform any additional controls, you should write an EXEC procedure to perform the control and invoke VMFLOAD from that EXEC procedure.

USING VMFLOAD

VMFLCAD searches the control file for the desired object mcdules in the order in which the identifiers are specified in the file, from top to

bottom. Remember that the updates were applied from bottom to top; the last update applied is the one immediately following the MACS record, and so on from top to bottom. For example, consider a control file, named LOC CNTRL, of the following format:

```
TEXT  MACS DMKMAC
LOCAL FIX1
SPEC  AUX1111
IBM1  AUXR20
```

In addition, suppose that the following object modules are on disk:

```
DMKACO     TEXT
DMKACO     TXTIBM1
DMKACO     TXTSPEC
DMKACO     TXTLOCAL
DMKPSA     TEXT
DMKSLC     TEXT
DMKMCH     TXTIBM1
DMKMCH     TEXT
DMKCKP     SPEC
DMKCKP     IBM1
DMKCKP     TEXT
```

And finally, assume that the loadlist called PARTIAL EXEC contains the following entries:

```
&1 &2 &3   DMKLD00E  LOADER
&1 &2 &3   DMKPSA
&1 &2 &3   DMKSLC
&1 &2 &3   DMKMCH
&1 &2 &3   DMKACO
&1 &2 &3   DMKCKP
```

When you issue the command VMFLOAD PARTIAL LOC, the VMFLOAD program punches the loader (DMKLD00E), and then punches the object modules in the following order:

```
DMKPSA     TEXT
DMKSLC     TEXT
DMKMCH     TXTIBM1
DMKVAT     TXTLOCAL
DMKCKP     TXTSPEC
```

The first file located that corresponds to the update level identifier LOCAL is punched, and all lower level files are ignored. If no file with a filetype of TXTLOCAL is found, VMFLOAD next searches for one with a filetype of TXTSPEC, then for one with a filetype of TXTIBM1, and lastly for a file with a filetype of TEXT. If the end is reached without finding a TEXT file, VMFLOAD displays the message:

```
'filename TEXT' NOT FOUND
```

and continues processing the next entry in the loadlist.

Thus, you can have a completed load file with object modules of varying levels. You are responsible for determining the validity of the completed load file.

The distributed system uses a loadlist named CPLOAD for the CP nucleus without virtual=real support, VRLOAD for the CP nucleus with virtual=real support, CMSLOAD for the CMS nucleus, and DMTLOAD for the RSCS nucleus. For example, to generate a CP nucleus, issue:

```
VMFLOAD CPLOAD DMKnnn
```

The loadlist, CPLOAD, contains first the loader, next all the nonpageable CP modules and lastly, the pageable CP modules. VMFLOAD first finds DMKLD00E (the loader) and punches it to the virtual punch. It then finds the highest levels of the other object modules named in CPLOAD (the highest level is determined by the control file, DMKnnn) and punches them to the virtual punch. If both a filename and a filetype are specified in the loadlist, the specified file is considered the highest level. If you issue the command "SPOOL PUN TO userid", the completed load deck is transferred to the virtual card reader of the userid specified. When you IPL that virtual card reader, the loader is read first, and it loads the rest of the object modules. If the loader is successful in its operation, the nucleus is written on the system residence device, if that device is available.

CP LOADLIST REQUIREMENTS

The CPLOAD loadlist EXEC contains a list of CP modules that is used by the VMFLOAD procedures to punch the text decks for the CP system. All modules following DMKCPE in the list are pageable CP modules. Each 4K page in this area may contain one or more modules. Pageable modules must not span the 4K page boundaries. The module grouping is governed by SPB (Set Page Boundary) cards that are assembled in the pageable modules. An SPB card is a loader control card that forces the loader to start this module at the next higher 4K boundary (a 12-2-9 multipunch must immediately precede the 'SPB' characters). If more than one module is to be contained in a 4K page, only the first is assembled with an SPB card. The second and subsequent modules for a multiple module 4K page must not contain SPB cards.

The loader inserts SPB cards automatically where they are needed; you do not have to insert SPB cards.

If you change the loadlist, be sure that any modules loaded together in the pageable area do not exceed the 4K limit. Page boundary crossover is not allowed in the pageable CP modules.

The position of two modules in the loadlist is critical. All modules following DMKCPE must be reenterable and must not contain any address constants referring to anything in the pageable CP area. DMKCKP must be the last module in the loadlist.

The Loader

The loader is a service program that loads a CP, CMS, or RSCS nucleus, and produces a load map. There are three loaders. DMKLD00E is the name of the CP loader, DMTLD00E is the name of the RSCS loader, and DMSLD00E is the name of the CMS loader. The CP and RSCS loaders are identical, except for the name. The CMS loader is similar to the CP loader; however, it contains only the function necessary to load CMS. The CP loader loads the object modules (TEXT files) supplied with it, resolves CCW addresses, and resolves address constants. The same loader is used whether a virtual=real or standard CP system is generated.

If overlay error occurs while the loader is executing, define a larger virtual machine and reload the system.

The loader is distributed with the following default I/O addresses:

```
Console=009
Printer=00E
```

You can override these addresses by placing a control card between the last card of the loader and the first card of the text decks. The format of the control card is:

Column	Contents
1	12-2-9 multipunch
2-4	DEV
5	blank
6-13	PRNT=cuu (cuu is the printer address)
14	blank
15-22	TYPW=cuu(cuu is the console address)

The other loader control statements are the same as the loader control statements described with the CMS LOAD command in the VM/370: Command Language Guide for General Users.

The loader is self-relocating, that is, it is initially loaded at address 2000 (hexadecimal); it then relocates itself at the top of storage. (For example, if the size of the loader is 10K, and the real storage size of the CPU is 512K, the loader occupies the area of storage between 502K and 512K.) As the loader needs free storage to perform its operations, it extends downward through storage.

The object modules being loaded must not overlay either the loader or any address between zero and 100 (hexadecimal). The object modules are loaded into storage in a positive direction (that is, upward through storage). Before the loader actually loads an object module, it checks that the module does not overlay the loader's free storage. If an object module would overlay the loader, the loader terminates. You must close the printer to get the load map printed. The last line of the load map indicates the overlay area, if there was one.

If the loader terminates the operation, one of the following wait conditions is indicated in the instruction counter:

Counter	Condition
X'111111'	A program check occurred
X'BBBBBB'	A machine check occurred
X'999999'	An SVC was issued

The interruption codes for an SVC wait state are:

Code	Meaning
100	Bad hexadecimal to binary conversion
101	No free storage remaining
102	Duplicate entry type 1 card.
103	The "name" in the LDT card is undefined.
104	Control section not found by EOF.
105	Reference table overflow.
106	Trying to overlay the loader.
107	Trying to overlay an address between zero and 100.
108	Permanent I/O error from the input device.
109	Loader releasing storage that is not on a doubleword boundary.

THE LOAD MAP

The load map (the output of the loader) indicates:

- The size of each object module and the address where it is loaded. For example:

```
DMKMCH AT 00E68   MODULE SIZE IS 000C00
```

- The end of the resident nucleus with the message:

```
***                                    ***
      END OF VM/370 RESIDENT NUCLEUS
***                                    ***
```

  The CP modules that precede this message in the load map are not pageable; the CP modules that follow this message are pageable.

- When a Set Page Boundary (SPB) card has been inserted. If an object module cannot fit on the same page as the object module(s) loaded before it, the loader inserts an SPB card to force the modules to be loaded at a page boundary. This procedure ensures that object modules do not cross page boundaries.

GENERATING A NEW LOADER

The loader service program, in its executable form, has a filetype of LOADER. Whenever you assemble a new copy of DMKLD00E, you must convert the resulting text file to a loader file. If there is a virtual punch at address 00D and a virtual reader at address 00C, the procedure for generating a new loader is:

STEP 1. ASSEMBLE THE NEW LOADER

Update and assemble DMKLD00E. The output from this assembly is DMKLD00E TEXT.

STEP 2. PUNCH A COPY OF THE OLD LOADER

Spool the punch continuously and punch a copy of the old loader.

```
SPOOL 00D * CONT
PUNCH DMKLD00E LOADER (NOH
```

STEP 3. PUNCH A COPY OF THE NEW LOADER TEXT FILE

Punch a copy of the newly-assembled loader, then close the punch. When the punch is closed the two files (DMKLD00E LOADER and DMKLD00E TEXT) are sent to your reader. The commands to punch the new loader text file and close the punch file are:

```
PUNCH DMKLD00E TEXT (NOH
SPOOL 00D * NOCONT CLOSE
```

STEP 4. LOAD THE NEW LOADER

IPL your virtual reader to read the old version of the loader (DMKLD00E LOADER) into your virtual machine. Then the old loader reads the new loader text file into your virtual machine and creates the new loader file.

```
IPL 00C
```

When the IPL is complete, the message

```
DMKDSP450W CP ENTERED; DISABLED WAIT PSW
```

is issued. The instruction address in the disabled wait PSW is X'404040'.

STEP 5. PUNCH A COPY OF THE NEW LOADER (EXECUTABLE FORM)

Close the punch to punch a copy of the new loader, which was created in Step 4.

```
CLOSE 00D
```

STEP 6. NAME THE NEW LOADER DMKLD00E LOADER

IPL CMS and read the file you punched in Step 5. Name this file DMKLD00E LOADER; this replaces the original DMKLD00E LOADER file with the new one.

```
IPL CMS
READ DMKLD00E LOADER
```

Note: Save a copy of the original DMKLD00E LOADER file before you replace it with the updated loader.

Using the GENERATE EXEC Procedure to Generate a New CP or CMS Nucleus

Use the GENERATE EXEC procedure to generate a new CP or CMS nucleus without performing the entire VM/370 system generation. GENERATE can:

- Build a new VM/370 directory.
- Assemble a new real I/O configuration file (DMKRIO).
- Assemble a new CP system control file (DMKSYS).
- Assemble a new forms control buffer load (DMKFCB).
- Assemble a new system name table (DMKSNT).
- Place a standalone version of the VM/370 service programs on disk. You can create a standalone version on disk of one or all of the following:

  --The Directory service program (DMKDIR)
  --The DASD Dump Restore service program (DMKDDR)
  --The Format/Allocate service program (DMKFMT)

- Create a standalone punch file of the VM/370 service programs. You can create a punch file for one or all of the following:

  --The Directory service program (DMKDIR)
  --The DASD Dump Restore service program (DMKDDR)
  --The Format/Allocate service program (DMKFMT)
  --The IBCDASDI Virtual Disk Initialization service program

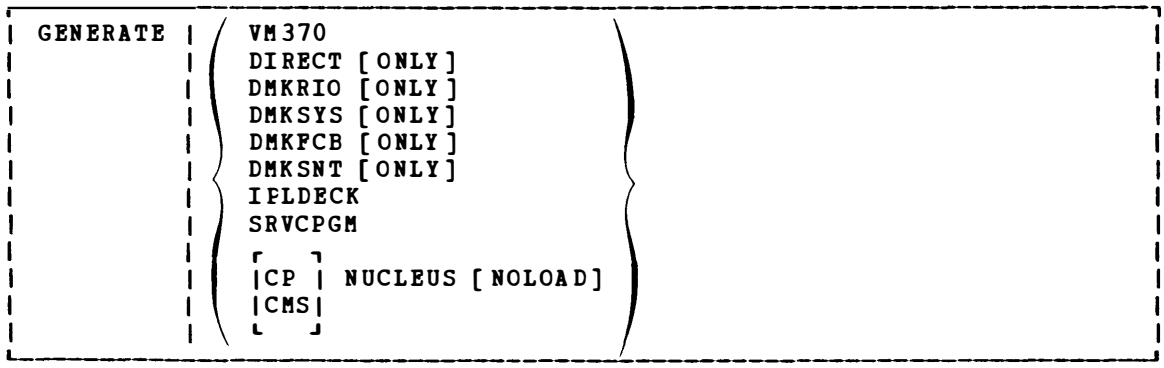
- Generate a CP and/or CMS nucleus and, optionally, load them.

Instructions for coding the control statements and macros that define your VM/370 directory, and DMKRIO, DMKSYS, and DMKSNT files are in "Part 2: Defining Your VM/370 System." Instructions for coding a new DMKFCB module are in the VM/370: System Programmer's Guide.

The GENERATE EXEC procedure assumes the following:

```
Virtual tape address = 182
Virtual address of CMS build area = 190
Virtual address of CP build area = 194
Virtual card reader = 00C
```

The format of the GENERATE EXEC command is:



where:

VM370 builds a new VM/370 directory, assembles DMKRIO and DMKSYS (also assembles DMKFCB and DMKSNT, if they are supplied), writes the CP nucleus to tape and optionally loads it.

You must have the appropriate files in your virtual card reader at address 00C. Place the following in your card reader, in the order shown:

```
:READ fn DIRECT
(Directory program control statements)
:READ DMKRIO ASSEMBLE
(real I/O configuration macros)
:READ DMKSYS ASSEMBLE
(CP system control macros)
```

where fn is the filename of your VM/370 directory. Optionally, you can place new versions of DMKFCB and/or DMKSNT in your card reader following the DMKSYS macros:

```
:READ DMKFCB ASSEMBLE
(forms control buffer load macros)
:READ DMKSNT ASSEMBLE
(system name table macros)
```

GENERATE reads the files in the reader. It invokes the Directory program to build the VM/370 directory and invokes the VMFASM EXEC procedure to assemble all the ASSEMBLE files in the reader. VMFASM is invoked with the current IBM supplied control file to ensure that the proper macro libraries are available when the modules are assembled and to assign the correct filetype. Then, GENERATE invokes the VMFLOAD EXEC procedure to place all the CP object modules on tape in the correct order. If an error is detected during any of these processing steps, GENERATE terminates at the end of that step.

For a CP nucleus without a virtual=real area, GENERATE loads the tape, thus loading the newly generated CP nucleus. For a CP nucleus with a virtual=real area, GENERATE writes the nucleus to tape and exits. You are instructed to shutdown the system. Then you can IPL the tape on a real machine or on a virtual machine that has enough virtual storage. The "Specifying a Virtual=Real Machine" section of Part 1 tells you how much virtual storage you need.

DIRECT [ONLY] builds a new VM/370 directory.

If you do not specify ONLY, GENERATE executes exactly as if you specified GENERATE VM370.

If you specify ONLY, only the VM/370 directory is built. You must place the Directory program control statements in your virtual card reader at address 00C, as follows:

```
:READ fn DIRECT
(Directory program control statements)
```

where fn is the filename of your VM/370 directory.

DMKRIO [ONLY] assembles the real I/O configuration file (DMKRIO) by invoking VMFASM.

If you do not specify ONLY, the GENERATE EXEC procedure executes just as if you specified GENERATE VM370 (except that it does not build a new VM/370 directory). Consequently, you should follow the directions for issuing GENERATE VM370, except do not place the Directory program control statement (nor the corresponding :READ statement) in your card reader.

If you do specify ONLY, only the DMKRIO module is assembled. You must place the real I/O configuration macros in your virtual card reader at 00C, as follows:

```
:READ DMKRIO ASSEMBLE
(real I/O configuration macros)
```

DMKSYS [ONLY] assembles the CP system control file (DMKSYS) by invoking VMFASM.

If you do not specify ONLY, the GENERATE EXEC procedure executes just as if you specified GENERATE VM370 (except that it does not build a new VM/370 directory and does not assemble a new DMKRIO). Consequently, you should follow the directions for issuing GENERATE VM370, except do not place the Directory program control statements or real I/O configuration macros (nor their corresponding :READ statements) in your card reader.

If you do specify ONLY, only the DMKSYS module is assembled. You must place the CP system control macros in your virtual card reader at 00C, as follows:

```
:READ DMKSYS ASSEMBLE
(CP system control macros)
```

DMKFCB [ONLY] assembles the forms control buffer load (DMKFCB) by invoking VMFASM.

If you do not specify ONLY, the GENERATE EXEC procedure executes just as if you specified GENERATE VM370 (except it does not build a new VM/370 directory and does not assemble new DMKRIO or DMKSYS modules. Consequently, you should follow the directions for issuing GENERATE VM370; except do not place the Directory control statements, real I/O configuration macros, or CP system control macros (nor their corresponding :READ statements) in your card reader.

If you specify ONLY, only the DMKFCB module is assembled. You must place the forms control buffer load macros in your virtual card reader at 00C, as follows:

```
:READ DMKFCB ASSEMBLE
(forms control buffer load macros)
```

DMKSNT [ONLY] assembles the system name table (DMKSNT) by invoking VMFASM.

If you do not specify ONLY, the GENERATE EXEC procedure goes on to invoke the VMFLOAD EXEC procedure to place all the CP object modules on tape. If no error occurs, and if the CP nucleus does not have a virtual=real area, GENERATE Then loads the tape, thus loading the CP nucleus (with the new version of DMKSNT).

If you do specify ONLY, the DMKSNT module is assembled, but the CP nucleus is not written to tape and is not loaded.

To assemble DMKSNT, you must place the system name table macros in your virtual card reader, at address 00C, as follows:

```
:READ DMKSNT ASSEMBLE
(system name table macros)
```

IPLDECK creates the standalone service programs on disk from their associated object modules (text decks). You are prompted to enter the names of the service programs you wish to generate. You can respond ALL, DDR, DIR, or FMT. If you respond ALL, the DASD Dump Restore, Directory, and Format/Allocate standalone programs are built on disk.

SRVCPGM punches all the standalone service programs (DMKDIR, DMKDDR, DMKFMT, and IBCDASDI).

[CP | CMS] NUCLEUS [NOLOAD] generates the CP or CMS nucleus. If you specify CP NUCLEUS, the CP nucleus is loaded onto tape.

For a CP nucleus without a virtual=real area, GENERATE loads the tape, thus loading the newly generated CP nucleus. For a CP nucleus with a virtual=real area, GENERATE writes the nucleus to tape and exits. You are instructed to shutdown the system. Then you can IPL the tape on a real machine or on a virtual machine that has enough virtual storage. The "Specifying a Virtual=Real Machine" section of Part 1 tells you how much virtual storage you need.

If you specify CP NUCLEUS NCLOAD, the tape is not loaded.

If you specify CMS NUCLEUS, a card-image deck is created and placed in the virtual card reader. GENERATE issues a prompting message to see if you want a card-image copy of the CMS nucleus put on disk. The card-image copy of the CMS nucleus is a file (CMSNUC NUCLEUS) that can later be loaded to create a CMS nucleus. If you specify CMS NUCLEUS NOLOAD, the new nucleus remains in the virtual card reader.

Using the VMFBLD EXEC Procedure to Build a New Nucleus

Use the VMFBLD EXEC procedure to build a new CP, CMS, or RSCS nucleus. The PLC (Program Level Change) tape must be mounted on a tape drive with an address of 181, and VMFBLD must be run from your C-disk. If these conditions are not satisfied, the following error messages are issued:

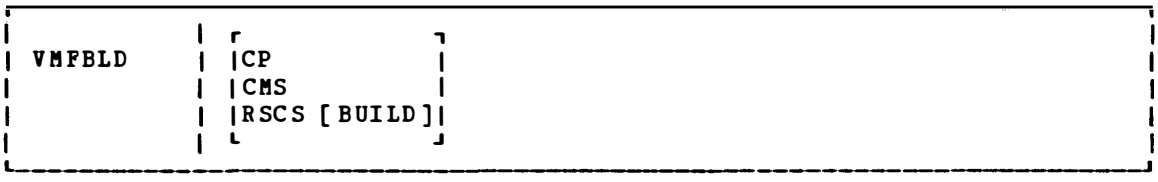
```
'VMFBLD EXEC' ** NOT RUNNING ON THE 'C' DISK
TAPE (181) NOT READY OR NOT ATTACHED *****
```

If you issue VMFBLD without an operand indicating which nucleus you want built, it prompts you with a message

```
ENTER (CP|CMS|RSCS [BUILD])
```

Then you must enter CP, CMS, or RSCS to build a nucleus.

The format of the VMFBLD command is:



where:

CP builds the CP nucleus, incorporating the changes on the PLC tape.

CMS builds the CMS nucleus, incorporating the changes on the PLC tape.

RSCS loads RSCS text decks onto staging area and gives you the option of building the RSCS nucleus.

RSCS BUILD
builds the RSCS nucleus. This operand assumes that RSCS text decks are already loaded on the staging area.

BUILDING THE CP NUCLEUS

To build the CP nucleus, you must log on the software system support virtual machine and IPL CMS. If you used the VM/370 Directory program control statements supplied with the starter system, you log on as MAINT. Then when you issue

```
VMFBLD CP
```

VMFBLD prompts you to enter the staging area address

```
ENTER CP STAGING AREA DISK ADDRESS:
```

If you are logged on as MAINT, the 194 minidisk is your staging area. If an error occurs, VMFBLD displays the message

```
ERROR ACCESSING SPECIFIED DISK
```

VMFBLD next prompts you with the message

```
ARE YOU A NEW USER?  - RESPOND (YES|NO)
```

You reply "yes" only when you are generating VM/370 from the starter system. The processing for the new user is described in "Part 3: Generating VM/370 (CP, CMS, and RSCS)". You reply "no" if you already generated VM/370 from the starter system and later want to apply PLC updates.

VMFBLD next asks:

```
VIRTUAL=REAL OPTION REQUIRED (YES,NO):
```

If you reply "yes", you are prompted to enter the size of your virtual=real area:

```
STORAGE SIZE OF VIRT=REAL [MINIMUM IS 32K]:
```

The minimum size virtual=real area is 32K and the maximum is 4M. If the size you specify is not on a page boundary, it is adjusted to a page boundary and you are notified with the message:

```
** SIZE ROUNDED TO NEXT HIGHER 4K BOUNDARY **
```

VMFBLD displays the size of the virtual=real area:

```
nnnnK STCRAGE SIZE FOR VIRTUAL=REAL
IS THE ABOVE ENTRY CORRECT (YES,NO):
```

If you reply "no", processing restarts by asking if you want the virtual=real option.

VMFBLD can build a tape that can be loaded. It issues the message:

```
GENERATE AN IPLABLE SYSTEM TAPE? -- RESPOND (YES|NO)
```

If you respond "yes", VMFBLD displays another message:

```
MOUNT TAPE AS 182, HIT CARRIAGE RETURN WHEN READY
```

You should mount a scratch tape on 182. If VMFBLD does not find a ready tape at 182, it issues the error message:

```
TAPE (182) NOT READY OR NOT ATTACHED *****
```

You must ready or attach tape drive 182 to continue.

VMFBLD can write the CP nucleus to disk, it prompts you:

```
WRITE THE NUCLEUS TO THE SYSTEM DISK? -- RESPOND (YES|NO)
```

If you respond "no", VMFBLD terminates. If you respond "yes", the following messages are displayed:

```
ROUTE LOAD MAP TO PRINTER OR READER? --RESPCND (RDR|PRT)
```

If you respond "rdr", VMFBLD finishes building the CP nucleus and issues the messages:

```
WHEN THE MESSAGE 'NUCLEUS LOADED ON 'xxxxxx' IS TYPED
ISSUE THE COMMANDS
CLOSE RDR ... (CLEARS THE READER)
CLOSE PRT ... (BRINGS THE LOAD MAP TO THE READER)
```

```
      SPOOL PRT OFF ... (RESETS PRINTER SPOOL ADDRESS)
      SHUTDOWN THE SYSTEM AND IPL THE NEW CP SYSTEM
```

If you respond "prt", VMFBLD finishes building the CP nucleus and issues the messages:

```
      WHEN THE MESSAGE 'NUCLEUS LOADED ON xxxxxx' IS TYPED
      ISSUE THE COMMANDS
         CLOSE RDR ... (CLEARS THE READER)
         CLOSE PRT ... (PRINTS THE LOAD MAP)
         SHUTDOWN THE SYSTEM AND IPL THE NEW CP SYSTEM
```

SPECIAL CONSIDERATIONS FOR GENERATING A CP NUCLEUS WITH A VIRTUAL=REAL AREA

If you generated a CP nucleus with a virtual=real area, you should check to see that there is enough virtual storage for you to IPL your nucleus in the virtual machine you are using. The "Specifying a Virtual=Real Machine" section of Part 1 tells you how to determine how much virtual storage you need. If you find that the software system support virtual machine that you are using does not have enough virtual storage, you should have VMFBLD generate a loadable CP nucleus on tape, but do not have VMFBLD write the nucleus to disk. To do this, reply "yes" to the question:

```
      GENERATE AN IPL'ABLE SYSTEM TAPE? -- RESPOND (YES|NO)
```

to create a loadable copy of CP on tape. Then respond "no" to the question:

```
      WRITE THE NUCLEUS TO THE SYSTEM DISK? -- RESPOND (YES|NO)
```

and VMFBLD terminates. Then you should shutdown the system and load the new nucleus on the real machine or on a virtual machine that has enough virtual storage.

BUILDING THE CMS NUCLEUS

To build the CMS nucleus, you must log on the software system support virtual machine and IPL CMS. If you used the VM/370 Directory program control statements supplied with the starter system, you log on as MAINT. Then you issue

```
      VMFBLD CMS
```

VMFBLD prompts you to enter the address of the CMS system disk with the message:

```
      ENTER CMS SYSTEM DISK ADDRESS:
```

If an error occurs, the message

```
      ERROR ACCESSING SPECIFIED DISK
```

is displayed.

VMFBLD can punch the standalone programs; it prompts you with the message

```
      PUNCH STANDALONE SERVICE PROGRAMS -- RESPOND (YES|NO)
```

If you reply "yes", VMFBLD punches the standalone programs and displays the following messages to indicate its progress:

```
PUNCHING 'IPL FMT'******
PUNCHING 'IPL DIR'******
PUNCHING 'IPL DDR'******
PUNCHING 'IPL IBCDASDI'******
IPL DECKS EXIST AS CLASS 'B' PUNCH FILES
```

If an error occurs punching the standalone service programs, VMFBLD displays the message:

```
****ERROR PUNCHING 'IPL xxx'****
```

VMFBLD continues by building the CMS nucleus. Also, if there are any updates on the PLC tape for the EREP program or the Assembler, VMFBLD updates those programs and creates the corresponding auxiliary directories. If an error occurs while VMFBLD is building the CMS nucleus, it issues one of the following messages:

```
***ERROR READING PLC TAPE OR WRITING TO DISK a***
***ERROR CREATING THE CMS NUCLEUS***
```

If the nucleus is built successfully, VMFBLD issues the message:

```
WHEN THE NEW CMS SYSTEM IS BUILT ISSUE:
   CLOSE PRT ... (PRINTS THE LOAD MAP)
```

BUILDING THE RSCS NUCLEUS

To build the RSCS nucleus, you must log on the software system support virtual machine and IPL CMS. If you used the VM/370 Directory program control statements supplied with the starter system, you log on as MAINT. Then issue:

```
VMFBLD RSCS [BUILD]
```

VMFBLD prompts you to enter the staging address:

```
ENTER CP STAGING DISK ADDRESS
194
```

If you are logged on as MAINT, the 194 minidisk is your staging area. If an error occurs, VMFBLD displays the message

```
ERROR ACCESSING SPECIFIED DISK
```

If you did not specify BUILD on the VMFBLD command line, VMFBLD loads the DMTSYS ASSEMBLE file and all the RSCS text files and prompts you:

```
DO YOU WISH TO BUILD RSCS SYSTEM -- RESPOND (YES|NO)
```

If you respond "no", VMFBLD processing is completed. If you respond "yes", processing continues as follows.

If you specified BUILD on the VMFBLD command line or responded "yes" to the preceding prompting message, VMFBLD prompts you:

```
ENTER RSCS SYSTEM DISK LINK PARAMETERS:
USERID VADDR1 VADDR2
rscs   191    195
```

You should reply with the userid of your RSCS virtual machine, the virtual address of the RSCS system disk in the RSCS virtual machine, and a virtual address with which MAINT can access the RSCS system disk. If you did not enter all the needed parameters or if VMFBLD was not able to link to the RSCS system disk, one of the following error messages is displayed:

```
MISSING PARAMETERS -- RE-ENTER
ERROR LINKING TO userid vaddr1 AS vaddr2
```

If VMFBLD gains access of the RSCS system disk, it copies the DMTNPT, DMTSML, DMTAXS, and DMTLAX TEXT files from the CP staging area disk to the RSCS system disk and displays the message:

```
TRANSFERRING 'RSCS' DISK RESIDENT TEXT .........
```

If an error occurs, VMFBLD displays the message

```
ERROR TRANSFERRING RSCS DISK RESIDENT TEXT FILE
```

If no errors occur while VMFBLD transfers the RSCS files, it builds the RSCS nucleus. If an error occurs building the RSCS nucleus, VMFBLD displays the message

```
*** ERROR CREATING THE RSCS NUCLEUS ***
```

Using the CMSGEND EXEC Procedure to Generate a CMS Module

Use the CMSGEND EXEC procedure to generate a new CMS module from a TEXT deck and place the new CMS module on the A-disk. Do not use CMSGEND to generate the System Assembler (use ASMGEND for the Assembler).

The format of the CMSGEND EXEC command is:

```
┌──────────────────────────────────────────────────────────────────────┐
│           │    ┌         ┐                                          │
│ CMSGEND   │ fn │ CTLCMS  │                                          │
│           │    │ CTLALL  │                                          │
│           │    │ NOCLEAR │                                          │
│           │    │ MAP     │                                          │
│           │    │ NOINV   │                                          │
│           │    └         ┘                                          │
└──────────────────────────────────────────────────────────────────────┘
```

where:

fn — is the filename of the TEXT file that is to be generated into a CMS module.

The filenames that may be specified in the CMSGEND command are: ACCESS, COMPARE, COPYFILE, CMSBATCH, CPEREP, DDR, DIRECT, EDIT, FILEDEF, FORMAT, GENDIRT, GEN3705, GLOBAL, LISTDS, LISTFILE, MACLIB, MODMAP, MOVEFILE, NCPDUMP, PRINT, PUNCH, QUERY, READCARD, RELEASE, RENAME, RENUM, SAVENCP, SET, SORT, SVCTRACE, SYNONYM, TAPE, TAPPDS, TEXTLIB, TYPE, UPDATE, VMFDUMP, VMFLOAD, and ZAP. The preceding filenames include CMS commands and service programs.

Note: You can also specify ASSEMBLE. When you specify ASSEMBLE, CMSGEND refreshes the Assembler's auxiliary directory. Use the ASMGEND EXEC procedure if you are updating assembler text decks or changing the mode.

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.

Responses

The CMSGEND EXEC procedure displays status and error messages.

```
*** CURRENT STATUS:
┌                              ┐
|'fn MODULE A2' EXISTS         |
|'fn MODULE A2' DOES NOT EXIST|
└                              ┘

┌                              ┐
|'fn MODOLD A1' EXISTS         |
|'fn MODOLD A1' DOES NOT EXIST|
└                              ┘
```

This message indicates whether a generated module already exists.

```
*** LOADING:
```

This message indicates that CMSGEND is loading the text decks.

```
*** (UNDEF. NAMES NORMAL FOR EDINIT)
*** NOW WE HAVE A SECOND PASS FOR EDINIT MODULE.
```

These messages indicate that the EDIT command requires two passes to resolve undefined names.

```
*** RESULTS:
['fn MODOLD A1' WAS ERASED]
['fn MODULE A2' RENAMED TO 'fn MODOLD A1']
 'fn MODULE A2' CREATED FROM TEXT DECK(S) ...
  WITH ATTRIBUTES ...
```

These messages indicate which existing modules were erased and renamed and tells which text decks were used to create the new module.

```
ERROR OCCURRED. CMSGEND STOPS.
```

This message indicates that an error occurred and that CMSGEND is terminated.

```
INVALID ARGUMENT fn
```

This message indicates that an invalid filename was specified on the command line.

Using the ASMGEND EXEC Procedure to Generate the Assembler

Use the ASMGEND EXEC procedure to build the System Assembler and to create the associated auxiliary directory. ASMGEND loads the text decks for the Assembler in the correct overlay structure and produces a load map. The format of the ASMGEND command is:

```
ASMGEND
```

Responses

The ASMGEND EXEC procedure displays the following status and error messages:

```
ENTER TARGET DISK MODE FOR ASSEMBLE MODULES
DEFAULTS TO S-DISK IF NONE ENTERED
```

You enter the mode letter of the disk containing the modules referred to from the auxiliary directory. 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.

```
ASMGEND XF GEND COMPLETE
```

This message indicates that the System Assembler and its associated auxiliary directory are generated successfully.

```
ASMGEND XF GEND FAILED
```

This message indicates that the System Assembler text files were not loaded successfully.

In order to preserve the integrity of VM/370 source and text files, you should keep updates and PTFs on a separate minidisk (not on the same disk as the original source and text files). This minidisk should contain the required IBM PTF updates from the latest PLC tape, updates that you make (such as expanding the accounting routines or adding a command to CP), and the resultant text files containing the updates. You need access to the latest macro libraries. If you used the VM/370 directory supplied with the starter system, the MAINT virtual machine has the CP library on the 194 minidisk and the CMS macro libraries on the 190 minidisk.

You can use the MAINT virtual machine described in the "CP/CMS PTF Application Procedures" section and illustrated in Figure 27 to update VM/370. Note that that virtual machine configuration consists of the MAINT entry in the IBM-supplied VM/370 directory, with the addition of MDISK statements for virtual disks (193, 294, 393, 394, and 390). Figure 27 shows the virtual disks described by the resultant MAINT entry. This virtual machine configuration should provide you with all the areas you need to update and test VM/370.

You should not change the source (ASSEMBLE) files directly but instead use the CMS UPDATE command and the VMFASM and/or the VMFMAC EXEC procedures supplied with the VM/370 system. Also, you should not change the IBM-supplied auxiliary files nor the PTF (WxxxxDMK) files as these are controlled by the PLC procedure.

If you want to update CP (for example, by adding your own command), you should create your own control file. This file should contain entries for your updates, your auxiliary files and your text names, and also all the IBM-supplied entries describing the IBM-supplied auxiliary files. The following is an example of a control file called YOUROWN CNTRL:

```
TEXT MACS DMKMAC USERLIB
LOCAL LCL
TEXT AUXR20
```

Suppose you wish to update the main processor for CP commands (DMKCPM) to add a module (DMKCMD) to process your new command. First you must update DMKCPM.

Log on as userid MAINT. Using the CMS EDIT command, with ASSEMBLE tab settings, create an update file. For this example, name the update file DMKCPM UPDTLCL.

After you create the update file, access the CP PTF disk, the CP text retention disk and the CP source disk. The CP text retention disk (194) should contain the latest level of DMKMAC MACLIB.

For example:

```
ACCESS 294 B/A
ACCESS 194 C/A
ACCESS 394 D/A
```

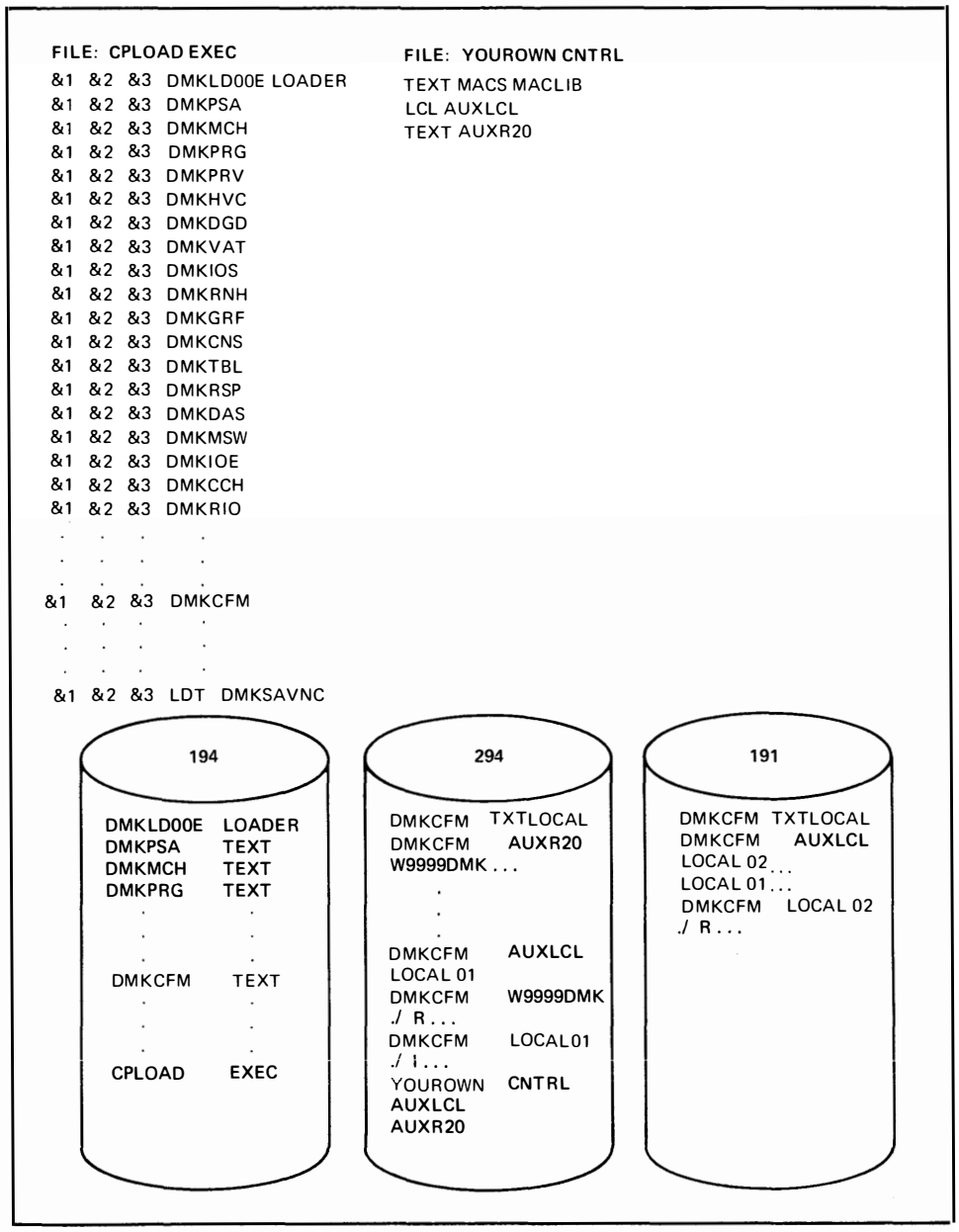


Figure 28. Files for a System Update

The CMS order of search is now:

```
191 A    R/W
294 B/A  R/O
194 C/A  R/O
394 D/A  R/O
190 S    R/O
```

Update the source file and assemble the new source of DMKCFM:

```
VMFASM DMKCFM YOUROWN
```

VMFASM updates the DMKCFM ASSEMBLE file with all the PTFs in the IBM-supplied auxiliary file for DMKCFM, and then updates it with your own update file (DMKCFM UPDTLCL). The updated source file is assemble and the text file is named DMKCFM TXTLOCAL.

If you have several of your own updates to apply, you can use an auxiliary file to apply your updates. For example, the control file YOUROWN CNTRL could be:

```
TEXT  MACS DMKMAC CMSLIB USERLIB
LOCAL AUXLCL
TEXT  AUXR20
```

Your auxiliary file contains a list of the filetypes of update files to be applied. For example, your auxiliary file (DMKCFM AUXLCL) might look like:

```
LOCAL02
LOCAL01
```

Then, your have two update files:

```
DMKCFM LOCAL01
DMKCFM LCCAL02
```

To update DMKCFM you must access the appropriate disks and issue the command:

```
VMFASM DMKCFM YOUROWN
```

VMFASM applies all the PTFs listed in the DMKCFM AUXR20 file and applies the two updates listed in your file (DMKCFM AUXLCL).

The IBM-supplied updates must be applied first; therefore, the entry in your control file identifying the IBM-supplied auxiliary file must be the last entry. Applying the IBM-supplied updates first ensures that the PTFs distributed by the VM/370 update service fit properly into the source files, as these are the same source files used by IBM.

After you update DMKCFM, you must reload the CP nucleus. You also want to add your new module to the CP modules. Assemble your module. Then edit the CP loadlist to include an entry for your module. Then punch a new CP nucleus by issuing:

```
VMFLOAD CPLOAD YOUROWN
```

BUILDING A CP NUCLEUS

The following is an example of building and loading a new CP nucleus. The MAINT virtual machine is used because it has enough storage for the loader to execute (the loader needs 512K of virtual storage). The MAINT virtual machine also contains the ECMODE option so that you can test the system after you load it onto a replica of the system residence disk (which should be attached to your virtual machine as the same address as the real system residence disk).

You now spool your punch to yourself so that the card deck can be placed in the virtual reader instead of being punched on the real system punch. Before punching the nucleus to your virtual card reader, you must make sure there are no files in the punch or reader. You then issue the VMFLOAD command specifying the CPLOAD EXEC file containing the list of modules to be loaded, and also specifying a control file (for example, YOUROWN CNTRL).

The sequence of commands and system responses is:

```
close pun
R;

close rdr
R;

pur rdr all
 NO FILES PURGED
R;

pur pun all
 NO FILES PURGED
R;

sp 00d *
R;
vmfload cpload yourown
SYSTEM LOAD DECK COMPLETE
PUN FILE 0821  TO  MAINT    COPY 01 NOHOLD
R;
```

The YOUROWN control file establishes both the filetype and order of search of the text files, and the CPLOAD EXEC establishes the filename. In Figure 28, note the reference to the DMKLD00E LOADER in the first line of the CPLOAD EXEC. Because the filetype of LOADER is specified, the control file is not used to determine the filetype. CPLOAD searches all disks until the DMKLD00E LOADER is found on the 194 disk. Then VMFLCAD punches the loader. Next, VMFLOAD searches for the DMKPSA TXTLOCAL text file. If it fails to find that text deck, VMFLOAD uses the next entry in the control file to determine the next possible filetype for DMKPSA. In this case, VMFLOAD searches for DMKPSA TEXT. When VMFLOAD finds the deck on the 194 disk, it punches it. This process is repeated for the next file, DMKMCH, and so on through the load list.

When VMFLOAD begins searching for the filename (DMKCFM), it finds a text deck called DMKCFM TXTLOCAL on the 191 disk. Note that a text with the same filename and filetype is also located on the 294 disk, but by accessing the 191 disk ahead of the 294 disk, a search priority is established.

The DMKCFM TXTLOCAL text deck found on the 191 disk is the result of applying the three updates that are on the 294 disk (which are also contained in the DMKCFM text deck on the 294 disk) and the newest update on the 191 disk, which the follcwing example tests.

You can test an updated text deck before replacing an older version of that text deck. Once the newer text deck is tested, you may replace the older one with the newer version.

Loading continues in a similar manner until all of the files specified in the CPLOAD EXEC are processed. At this time, the "SYSTEM LOAD DECK COMPLETE" message is displayed, and, as the punch is closed, the punch transfer message is also displayed.

At this point the standalone loader is in the card reader, followed by all of the text decks necessary to construct a CP nucleus. There are two ways to handle this reader file.

First, you can use the CMS MOVEFILE command to place the entire file on tape, thus creating a CP nucleus load tape which can be loaded into real storage. However, remember that the loader requires 512K of real storage.

The second way to handle the CP nucleus reader file is to IPL the loader in the card reader and actually load the nucleus onto the 330 virtual disk, which is an exact copy of the real system residence device. To do this, issue the CP IPL command and specify a unit address of 00C. When the load operation completes, the message "NUCLEUS LOADED ON SYSRES" is displayed, followed by an error message indicating a disabled wait state PSW, the normal termination of the standalone loader program.

```
ipl 00c
NUCLEUS LOADED ON SYSRES
DMKDSP450W CP ENTERED; DISABLED WAIT PSW
CP
```

Now, you must define your console address to be the same as defined in the RIOGEN macro in DMKRIO. Then IPL the system residence device, which is a virtual disk with an address of 330. (This procedure verifies that the nucleus runs on a virtual machine; this procedure is recommended, although it is not required.)

```
def 009 as cuu
CONS cuu DEFINED
ipl 330
```

The following example shows the new nucleus being loaded. The two error messages (both DMKLNK108E) are the result of not defining those disks as addresses which are contained in the virtual system's DMKRIO (Real I/O) deck.

```
VM/370 VERSION 2 LEVEL 0

NOW 14:53:07 EST THURSDAY 03/28/74
CHANGE TOD CLOCK (YES|NO) :no
14:53:50 DMKLNK108E CMSSYS 190 NOT LINKED; VOLID UDISK2 NOT MOUNTED
RRRR....RING....GGGG
14:53:50 DMKLNK108E OPERATOR 191 NOT LINKED; VOLID UDISK1 NOT MOUNTED
RRRR....RING....GGGG
14:53:50 START ((COLD|WARM) (DRAIN))|(SHUTDOWN) : shutdown
14:53:50 LOGON AT 14:53:50 EST THURSDAY 03/28/74
14:53:50 LINE 01F LOGON AS OPERATOR USERS = 001

DMKCPI960W SYSTEM WARM START DATA SAVED

DMKCPI961W SYSTEM SHUTDOWN COMPLETE

DMKDSP450W CP ENTERED; DISABLED WAIT PSW
CP
```

After you check the new CP, you may redefine your console and IPL the CMS system. After you IPL CMS, enter the DDR command and create a backup copy of the CP nucleus (which can be restored to the real system). Define your input unit as the address of your system residence device. Your output unit is the tape (181) that is attached to your virtual machine. When you enter the DUMP statement with the NUCLEUS operand, DDR creates a copy of the nucleus that was just loaded. This copy may later be restored using the standalone version of the DDR program on the real machine, and it also provides a backup copy of the nucleus.

The sequence of commands and responses is:

```
def cuu as 009
CONS 009 DEFINED
ipl cms
CMS...mm/dd/yy
ddr
ENTER: in 330 3330 sysres
ENTER: out 181 2400
ENTER: dump nuc
DUMPING SYSRES
END OF DUMP
ENTER:
END OF JOB
R; T=0.21/2.63 15:04:04
```

You created a backup copy of the CP nucleus.

BUILDING A CMS NUCLEUS

The same procedure can be used to update CMS modules. In this case, the order of search for updating is:

```
191 A   R/W
293 B/A R/O
190 C/A R/O
393 D/A R/O
```

To test CMS in a virtual machine before updating the real CMS system disk, adequate disk space must be available to contain a copy of the nucleus being tested (for example, on the virtual disk 390 as shown in Figure 27.) You can test modules on your 191 disk. Note that when disks are searched for a disk-resident command, the normal order of search is followed. Thus, if you have a module on your virtual 191 disk with the same filename as one on the system disk, the module on your 191 disk is found first (assuming your 191 disk is accessed before your system disk). Again, the userid MAINT can be used.

To build a new CMS nucleus, you must:

- Create a new CMS system disk.
- Generate a new CMS nucleus.
- Optionally, save the new CMS operating system.

CREATING A CMS SYSTEM DISK

Use the following procedure to create a new CMS system disk:

- IPL 190.
- Issue the CMS FORMAT command to format the minidisk that you want to contain the new CMS operating system.
- Issue the CMS FORMAT command with the RECOMP option to format the same minidisk with one or two cylinders less than the total number of cylinders on the disk (one on a 3330, two on a 2314 or 3340). The last cylinder(s) are used for the CMS nucleus.
- Regenerate any MODULE file that uses an auxiliary directory. Auxiliary directories are described in the VM/370: System Programmer's Guide. Two modules distributed with the CMS system, ASSEMBLE and CPEREP, use an auxiliary directory. Regenerate ASSEMBLE by invoking the ASMGEND EXEC procedure and regenerate CPEREP by invoking CMSGEND. Both of these procedures are described earlier in Part 6. IBM Program Products may also use an auxiliary directory.
- Generate a new CMS nucleus using the sequence of commands described in the following section.

GENERATING THE CMS NUCLEUS

Use the following commands to generate a new CMS nucleus:

- Load the CMS system disk if you have not already loaded it. The address is usually 190.

```
ipl cuu
```

- Access the disk that contains the CMS text files that are used to create the nucleus.

```
access cuu a
```

- Spool your punch to your reader.

```
spool d to *
```

- Punch a copy of the CMS nucleus.

```
vmfload cmsload dms
```

- Load the CMS nucleus into storage. A load map is produced and spooled to the virtual printer.

```
ipl c
```

After the IPL sequence, the following questions are asked.

DMSINI606R SYSTEM DISK ADDRESS = cuu

Enter the device address (cuu) of the system disk (S-disk). On this disk CMS expects to find all CMS system information and programs not contained within the CMS nucleus, such as the disk-resident command modules. If the CMS nucleus is

written on this disk, then cuu is also the IPL device address.

If you enter an invalid device address, the message

DMSINI079E INVALID DEVICE ADDRESS - REENTER

is issued. Message DMSINI606R is reissued so that you can enter a valid device address.

If you press the carrier return without entering a device address, X'190' is assumed to be the system disk address.

Once the system disk address entered is accepted, message DMSINI615R is issued.

DMSINI615R Y-DISK ADDRESS = cuu

Enter the device address (cuu) of the system disk extension (Y-disk). On this disk CMS expects to find all CMS system information and programs not contained within the CMS nucleus and not on the S-disk. If the CMS nucleus is written on the Y-disk, then cuu is also the IPL device address.

If you enter an invalid device address, the message:

DMSINI079E INVALID DEVICE ADDRESS - REENTER

is issued. Message DMSINI615R is reissued so that you can enter a valid device address.

If you press the carrier return without entering a device address, X'19E' is assumed to be the address of the system disk extension.

Note: If you do not want to have a Y-disk, do not attach the device that was specified (or defaulted to) as the Y-disk address.

Once the address entered for the system disk extension is accepted, message DMSINI607R is issued.

DMSINI607R REWRITE THE NUCLEUS? (YES|NO)

If you enter "yes", a copy of the CMS nucleus is written onto the disk indicated in the response to message DMSINI608R. If you enter "no", the CMS nucleus is not written to disk.

If you enter neither "yes" nor "no," the message

DMSINI081E INVALID REPLY - ANSWER "YES" OR "NO"

is issued. Message DMSINI607R is reissued so that you can enter a valid response.

If you enter "no", the remaining questions in generating a new CMS nucleus are skipped and control is passed to the CMS initialization routine.

If you enter "yes", message DMSINI608R is issued.

DMSINI608R IPL DEVICE ADDRESS = cuu

Enter the address of the device (cuu) on which the CMS nucleus is to be written. If the system disk and the IPL device are to be the same, you need only press the carrier return.

If you enter an invalid device address, the message

DMSINI079E INVALID DEVICE ADDRESS - REENTER

is issued. Message DMSINI608R is reissued so that you can enter a valid device address.

If the IPL device you designated is not currently defined, is not in read/write status, or is an unsupported device type, the message

DMSINI082E IPL DEVICE ERROR - REENTER

is issued. Message DMSINI608R is then reissued. At this time, you may enter CP mode by pressing the Attention key (or equivalent), then determine the status of the device you designated by entering the CP command

QUERY VIRTUAL cuu

and take the corrective action necessary to define the device for your virtual machine or to access it in read/write status. You may reenter CMS by issuing the CP command

BEGIN

Then you must reenter the device address. Once the device address is accepted, message DMSINI609R is issued.

DMSINI609R NUCLEUS CYL ADDRESS = nnn

Enter the cylinder number (nnn), for the device entered in response to message DMSINI608R, where the CMS nucleus is to be written. The number (nnn) must be between 001 and m-1 (where m equals the number of cylinders on the disk, the cylinders being numbered from 0 to m-1). The number, nnn, must be entered in decimal. If the system disk and the IPL device are the same, follow the directions for issuing the FORMAT command in the "Creating a CMS System Disk" section.

If you do not enter a valid decimal cylinder number, the message

DMSINI080E INVALID CYLINDER NUMBER - REENTER

is issued. Message DMSINI609R is reissued and you may enter a valid cylinder number.

If the cylinder specified is not greater than the number of cylinders already in use on the device (as indicated in the Master File Directory, then the message

DMSINI083E NUCLEUS WILL OVERLAY FILES - RECOMPUTE

is issued. You may respond with a larger cylinder number, or IPL CMS and format the specified IPL device with the RECOMP option.

Once the nucleus cylinder address is accepted, message DMSINI610R is issued.

DMSINI610R ALSO IPL CYLINDER 0? (YES|NO)

The initial IPL text is always written on the same cylinder as the CMS nucleus (the cylinder designated in response to message DMSINI609R). The initial IPL text is a bootstrap program which reads the nucleus from the designated cylinder. If it is not also written on cylinder 0, then you must enter the cylinder number when subsequent IPL commands are issued for the system being generated. See the IPL command description in the VM/370: Command Language Guide for General Users. Your response has the following meaning:

yes Initial IPL text is written on cylinder 0 as well as on the cylinder designated in response to message DMSINI609R.

no Initial IPL text is written only on the cylinder designated in response to message DMSINI609R.

If you do not enter "yes" or "no," the message

DMSINI081E INVALID REPLY - ANSWER "YES" OR "NO"

is issued. Message DMSINI610R is reissued so that you can enter a valid response.

If your response is valid, message DMSINI611R is issued.

DMSINI611R VERSION IDENTIFICATION =

Enter up to 32 bytes of information, including blanks, to specifically identify the version and level of CMS; this information is printed each time you IPL the CMS system now being generated. The default identification (specified by a carrier return) is:

CMS VERSION n.n - mm/dd/yy

where n.n is the version and level of CMS, and mm/dd/yy is the month, day and year the CMS nucleus was created.

DMSINI612R INSTALLATION HEADING =

Enter up to 64 bytes of informaticn, including blanks, to serve an installation standard heading at the beginning of each output file. The default heading (specified by a carrier return) is:

CONVERSATIONAL MONITOR SYSTEM

The nucleus is then written on the specified disk cylinder and the version identification is displayed, indicating that the CMS system is loaded successfully and is ready to accept CMS commands.

At this point, you may save the CMS system. See the section on "Saved Systems." If you wish to save a copy of the nucleus as a disk file, which can be punched to the virtual punch later, you should issue SPOOL C HOLD so that a copy of the nucleus remains in the card reader after it

is loaded. Also, if you wish to save a copy of the load map as a disk file, spool your printer to yourself.

The following is an example of the commands you issue to load the new CMS nucleus:

- Direct the nucleus to your reader.

```
spool d to *
R;
```

- Punch the nucleus.

```
vmfload cmsload dms
SYSTEM LOAD DECK COMPLETE
PUN FILE 0355 TO MAINT COPY 01 NOHOLD
R;
```

- Hold the file in the reader so that it is not lost after the IPL completes.

```
spool c hold
R;
```

- Spool the printer to your reader.

```
spool e to *
R;
```

- IPL the card reader, thereby creating the load map.

```
ipl c

DMSINI696R SYSTEM DISK ADDRESS = 130
DMSINI615R Y-DISK ADDRESS = 19e
DMSINI607R REWRITE NUCLEUS ? yes
DMSINI608R IPL DEVICE ADDRESS = 191
DMSINI609R NUCLEUS CYL ADDRESS = 9
DMSINI610R ALSO IPL CYLINDER 0 ? yes
DMSINI611R VERSION IDENTIFICATION =
DMSINI612R INSTALLATION HEADING =
CMS VERSION 2.0 - 03/29/74 16:47
R;

spool rdr nohold
close prt
close rdr
PRT FILE 0342 TO MAINT COPY 01 NOHOLD
```

- Read a copy of the CMS nucleus to disk.

```
read cmsnuc nucleus a1
R;
```

- Read a copy of the CMS load map to disk.

```
read cmsnuc loadmap a1
RECORD LENGTH IS '132' BYTES.
R;
```

You now have two CMS files on this 191 disk: CMSNUC NUCLEUS, which contains the CMS nucleus created above, and CMSNUC LOADMAP, the load map for this nucleus. This procedure is useful when you want to punch the file containing the particular nucleus created to a virtual card reader at some later time.

To regenerate a nucleus which exists as a disk file (CMSNUC NUCLEUS, for example), issue the following commands:

```
spool pun to *
punch cmsnuc nucleus a1 (noheader)
ipl 00c
```

You may then answer the questions previously described.

If you wish, you can test a new CMS nucleus by writing the test nucleus on device 390. To do this, respond as follows to these two questions:

```
DMSINI606R SYSTEM DISK ADDRESS = 190
DMSINI608R IPL DEVICE ADDRESS = 390
```

By specifying different devices as your S-disk and IPL device, you can test the updated CMS by loading 390 before putting the updated CMS nucleus into general use.

CMS modules can be tested on your virtual 191 before they are placed on the system disk.

Once the update is successfully tested, you can move it from your 191 to your 193 for permanent retention.

SAVING THE CMS SYSTEM

The DMKSNT module must have been configured when CP was generated. If you coded a NAMESYS macro for CMS, you have an entry in the system name table and can save the CMS system by entering the SAVESYS command as the first command after IPL. Enter the command:

```
SAVESYS name
```

as the first command after the IPL command. Enter the SAVESYS command when the CMS version identification is displayed. The "name" specified on the SAVESYS command is the name assigned to the saved system via the NAMESYS macro.

When you IPL a saved CMS system, CMS operates as if an IPL of a specific device occurred, with the single exception that the directory for the system disk is part of the nucleus.

CMS SOURCE PRCGRAMS

To format a minidisk to contain the CMS source programs, IPL the CMS system disk and issue the CMS FORMAT command. This minidisk must be at least 120 cylinders for a 2314 (or 2319), 60 cylinders for a 3330 disk, or 160 cylinders for a 3340 disk. Then, you should have the CMS source tape mounted and attached to the virtual machine, and issue the following commands to load the source programs onto the CMS disk:

```
TAPE FSF
TAPE LOAD (EOF 2)
```

CREATING CMS DISK-RESIDENT MODULES

CMSGEND is a procedure provided to create CMS disk-resident command modules from CMS TEXT files. The procedure is invoked by specifying:

```
|                        ┌       ┐
|     CMSGEND filename   |CTLCMS |
|                        |CTLALL |
|                        |NOCLEAR|
|                        |MAP    |
|                        |NOINV  |
|                        └       ┘
```

where filename is the name of the module to be generated.

CMSGEND creates a history of modules. Any existing file with the identification of 'filename MODULE A2' is renamed 'filename MODOLD A1'. An existing file of 'filename MODOLD A1' is erased. The CMSGEND EXEC procedure displays the "CURRENT STATUS" of these modules.

CMSGEND then loads the TEXT files that comprise the module 'filename'. A "LOADING" message is displayed, and 'filename MODULE A2' is generated. The CMSGEND EXEC procedure then displays the results of the procedure, including the identification of the command and the text files used to generate it.

You must access the S-disk as your read/write A-disk, and have all pertinent TEXT files available before invoking the CMSGEND EXEC procedure. For example, to replace an ACCESS module, enter:

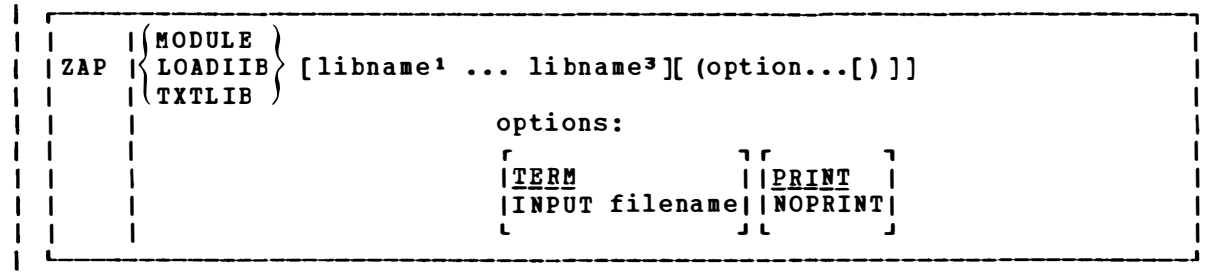
```
cmsgend access

*** CURRENT STATUS:
FILE 'ACCESS MODULE A2' DOES NOT EXIST
FILE 'ACCESS MODOLD A1' DOES NOT EXIST
*** LOADING:
 ACCESS   SD 00E000
 READFST  SD 00EAC8
 INVALID CARD - V0142DMS ELIMINATES WRITE BACK OF MED TO FIND OUT IF
                   DISK IS R/O
 INVALID CARD - *        DMSACM   V0142DMS  D1 CMS191  4/18/74 14:47
 INVALID CARD - *        CMSLIB   MACLIB    D1 CMS191  4/21/74  8:44
 INVALID CARD - *        OSMACRO  MACLIB    S2 CMS190  4/22/74 10:15
 INVALID CARD - *        DMSACM   ASSEMBLE  C1 SOURCE  4/18/74 17:29
 DMSACM   SD 00EDB0
 READMFD     00EDB0
 DMSALU   SD 00F0A8
 RELUFD      00F0A8
 END$RELU    00F1F8

*** RESULTS:
' ACCESS MODULE A2' CREATED FROM TEXT DECK ( S ) DMSACC DMSACF DMSACM DMSALU
WITH ATTRIBUTES TRANS NOMAP
R;
```

USING THE ZAP SERVICE PROGRAM TO UPDATE CMS MODULE, LOADLIB, OR TXTLIB FILES

ZAP is a CMS command that modifies or dumps MODULE, LOADLIB, or TXTLIB 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, or an entire member of a LOADLIB or TXTLIB file, or an entire module of a MODULE file.

The format of the ZAP command is:



where:

MODULE
LOADLIB
TXTLIB
: indicates the type of file that is to be modified or dumped.

libname
: is the library name containing the member to be modified or dumped. You can specify one to three library names. The libname 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. All output except control records displays at the terminal.

INPUT filename [PRINT | NOPRINT]
: specifies that input is submitted from a disk file, 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 and control records in error and error messages display at the terminal.

If you specify NOPRINT with INPUT filename, nothing prints on the printer. All output displays at the terminal.

The following table shows the resulting output of valid option combinations:

OPTIONS	PRINT	NOPRINT
INPUT	Commands and control records in error and error messages on the terminal. Everything to printer.	Everything on the terminal. Nothing on the printer.
TERM	Only error messages on the terminal. Everything on the printer.	Everything except control records on the terminal. Nothing on the printer.

ZAP INPUT CONTROL RECORDS

Seven types of ZAP control records exist: NAME, DUMP, BASE, VER or VERIFY, REP, comment, and END.

ZAP control records are free form and need not start in position one of the record but the ZAP program can accept only 80 characters of data for each control 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 (0D, 02C8, 014318). Data fields in VER and REP control records must also contain an even number of hexadecimal digits, but are not limited to six digits.

If you wish, you may separate the data anywhere by commas (for example, 83256482 or 8325,6482). The commas have no effect on the operation.

The program sets the NOGO switch on if a control record is found to be in error. A file cannot be modified once 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 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.

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. The format of the DUMP control record is:

```
DUMP { membername } [ csectname [startaddress [endaddress]] ]
     { modulename }  [ ALL                                   ]
```

where:

membername is the name of the member to be dumped, or the member that contains the CSECT(s) to be dumped. This member must be found in one of the libraries specified in the ZAP command line. However, if the library is a CMS TXTLIB, its directory does not contain member names. Therefore, the program ignores the member name (although you must specify it), and the program searches for the csectname (which you must specify).

modulename is the name of the module to be dumped, or the module that contains the CSECT(s) to be dumped. 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 is to be dumped. If you do not specify csectname, the program dumps only the first CSECT.

The csectname is required for CMS TXTLIBs, optional for OS TXTLIBs, LOADLIBs, and MODULE files. (See the discussion of csectname under "Name Control Record.")

You must not specify csectname for a module created with the NOMAP option.

ALL specifies to the program to dump all CSECTs within the specified member or module. You can specify ALL for MODULE files, LOADLIBs, and OS TEXTLIBs, but not for CMS TXTLIBS. If you wish to dump all the CSECTs in a member of a CMS TXTLIB, you must issue a separate DUMP control record for each CSECT.

startaddress is the location within the specified CSECT where the dump is 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, if you wish to start dumping at address 08 in a CSECT that begins at location 400, you specify start address or 08, not 0408.

endaddress is the last address to be dumped. This must be two, four, or six hexadecimal digits. If you specify no address, the program dumps the rest of the CSECT. Note that start and end addresses apply only when you specify a csectname.

If the file to be dumped contains undefined areas (such as a DS in a TXTLIB member), the hexadecimal portion of the dump contains blanks to indicate that the corresponding positions are undefined.

NAME Control Record

The NAME control record specifies the member or module and CSECT that contain the data to be verified or replaced by the ZAP operation. The format of the NAME control record is:

```
NAME { membername } [csectname]
     { modulename }
```

where:

{membername / modulename}
is the member or module that you want to be searched for the desired CSECT.

csectname is the name of the desired control section. You must specify csectname if the CSECT you wish to modify is in a CMS TXTLIB (that is, 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. The CSECT name specified in the NAME record is compared with CSECT names in the directory. If a CSECT match is found and no member name match is found, the member selected is the one that contains the CSECT name.

The csectname is optional if the CSECT you wish to modify 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.

The csectname is optional for a MODULE file. The module named in the NAME control record is located and, if you specified 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 the csectname.

If you do not specify csectname, the program uses the beginning location of the module. You are not allowed to

specify csectname if the module was created with the NOMAP option.

The NAME control record must precede the BASE, VER, and REP control records. If it does not, the program sets the NOGO switch on.

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 format of the BASE control record is:

```
┌──────────────────────────────────────────────────────────────┐
│                                                              │
│  BASE    address                                             │
│                                                              │
└──────────────────────────────────────────────────────────────┘
```

where:

address is the starting address of the CSECT. The address must be two, four, or six hexadecimal digits.

For example, for a CSECT starting at location 400, you would specify the BASE 0400 in the BASE control record. If a subsequent VER card requests verification of location 0408, the BASE of 0400 is subtracted from 0408, and the program verifies location 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 performed as if the BASE address is location 0. For example, if you specify a BASE of 400 and the address you wish to inspect or modify is 408, then you must specify 08 and not 408 in REP and VER control records. The address in this case is from the start of the module. If you do not specify csectname in the NAME control record, you cannot specify any BASE value other than 00.

The BASE control record is optional. See the discussion under "VER or VERIFY Control Record." If specified, the BASE control record 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.

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 perform a subsequent REP operation until it encounters another NAME control record.

The VER control record is optional. More than one VER record can follow a single NAME record.

The format of the VER control record is:

where:

disp — is the hexadecimal displacement of the data to be inspected from the start of the CSECT, if you did not submit a BASE control record for this CSECT. If you did submit a BASE control record, then disp is the actual location of the data. The 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 to be compared. This must be an even number of hexadecimal digits.

For example, if the location you wish to verify is 3CC, and the CSECT begins at location 2B0, you can either issue:

```
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 issue:

```
VER 011C data
```

This also applies to the disp operand of the REP control record.

REP Control Record

The REP control record modifies instructions or data at the specified location within the CSECT that you specified 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 perform additional REP operations until it encounters another NAME control record.

The REP control record is optional. More than one REP record can follow a single NAME record.

The format of the REP control record is:

```
┌──────────────────────────────────────────────────────────────┐
│                                                              │
│  REP    disp    data                                         │
│                                                              │
└──────────────────────────────────────────────────────────────┘
```

where:

disp — is the hexadecimal displacement of the data to be replaced from the start of the CSECT, if you did not submit a BASE control record for this CSECT. If you did submit a BASE control record, then disp is the actual location of the data. The 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 modified, the program does not perform 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.

Note: 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.

Comment Control Record

The ZAP program ignores comment control records. If the PRINT option is in effect, the program prints the comments. The format of a comment record is:

```
┌──────────────────────────────────────────────────────────────┐
│                                                              │
│  *  comment                                                  │
│                                                              │
└──────────────────────────────────────────────────────────────┘
```

You must follow the asterisk with at least one blank.

END Control Record

The END control record ends ZAP processing. The END record is required and must be the last control record. The format of the END control record is:

```
┌──────────────────────────────────────────────────────────────┐
│                                                              │
│  END                                                         │
│                                                              │
└──────────────────────────────────────────────────────────────┘
```

SPECIAL CONSIDERATIONS FOR USING THE ZAP SERVICE PROGRAM

Before you use 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 modified returns to its original status.

If you issue REP control record against a file that contains an undefined area (for example, a Define Storage area) within the REP data field and do not issue a VER control record prior to the REP control record, the bytes prior to the undefined area, if any, are modified and all the bytes after the undefined area are not modified. The program prints warning message DMSZAP248W.

BUILDING AN RSCS NUCLEUS

The same procedure used to update CP and CMS can be used to update RSCS. However, unlike CP and CMS, RSCS can test the system that is built; it does not need to test a duplicate copy of the system that is built. In this case, the order of search for updating is:

```
195 A
194 B/A
```

Again, the MAINT virtual machine can be used to do the updating.

To build a new RSCS nucleus you must create a new RSCS system disk and generate a new RSCS nucleus.

CREATING AN RSCS SYSTEM DISK

Use the following procedure to create a new RSCS system disk:

- Log on as MAINT.
- IPL 190.
- Link to the minidisk that you want to contain the new RSCS nucleus as 195 in write status and access it as your A-disk.
- Issue the CMS FORMAT command to format that minidisk.
- Issue the CMS FORMAT command with the RECOMP option to format the same minidisk with one or two cylinders less than the total number of cylinders on the disk (one less on a 3330, two less on a 2314 or 3340). The last cylinders are used for the RSCS nucleus.
- If you wish to change the RSCS configuration, recreate the AXSLINKS, LAXLINES, and TAGQUEUE COPY files and create a new DMTLOC macro library. See "Part 3: Generating VM/370 (CP, CMS, and RSCS)."
- Generate a new RSCS nucleus using the commands described in the following section.

- Load the RSCS files onto the CP system disk (the 194 minidisk belonging to MAINT) if they are not already there.

- Access the MAINT 194 disk as an extension of the RSCS system disk.

```
access 194 b/a
```

  Note: If you want to apply your own updates, they should be on the 294 disk. Then disk access should be:

```
access 294 b/a
access 194 c/a
```

- Assemble the RSCS configuration table module, DMTSYS, using the VMFASM EXEC procedure. The control file DMTR20 identifies DMTLOC as the macro library.

```
vmfasm dmtsys dmtr20
```

  If an error occurs due to an incorrectly coded macro, correct the macro and restart by generating a new DMTLOC macro library.

  Note: If you do not have enough disk space to assemble, acquire additional T-disk space.

- Close and clear the punch and reader. Spool the punch to your virtual card reader. Use the VMFLOAD EXEC procedure to punch the RSCS nucleus. The loadlist EXEC, DMTLOAD, contains the filenames of all the RSCS TEXT modules.

```
close pun
R;
close rdr
R;
purge rdr all
R;
purge pun all
R;
spool pun to *
R;
vmfload dmtload dmtr20
SYSTEM LOAD DECK COMPLETE
PUN FILE spoolid TO userid
R;
```

- Copy the RSCS Supervisor TEXT decks, DMTAXS and DMTLAX, from the MAINT 194 disk to your RSCS system disk.

```
copyfile dmtaxs text b1 = =  a1 (replace olddate
R;
copyfile dmtlax text b1 = = a1 (replace olddate
R;
```

- Copy the required line driver TEXT decks, DMTNPT and/or DMTSML, from the MAINT 194 disk to the RSCS system disk. The MAINT 194 disk can now be released and detached.

```
copyfile dmtnpt text b1 = = a1 (replace olddate
R;
copyfile dmtsml text b1 = = a1 (replace olddate
R;
release 194
R;
detach 194
DEV 194 DETACHED
R;
```

- IPL your card reader which contains the RSCS nucleus that was punched in Step 8. A series of prompting messages requests information relative to the physical location and disk address of the RSCS nucleus. Answer them as shown.

```
ipl c
DMTINI407R REWRITE THE NUCLEUS ? yes
DMTINI408R IPL DEVICE ADDRESS = 195
DMTINI409R NUCLEUS CYL ADDRESS = 003 (or, 004 for 3330)
DMTINI410R ALSO IPL CYLINDER 0 ? yes
```

The message

```
DMTAXS103E FILE 'spoolid' REJECTED -- INVALID DESTINATION
           ADDRESS
```

is issued. It indicates that the RSCS nucleus file was purged from the card reader.

- The message shown in line 1 of the following example indicates completion of the writing of the RSCS nucleus on the specified disk. IPL the specified disk and start your RSCS operations as described in the VM/370: Remote Spooling Communications Subsystem (RSCS) User's Guide.

```
DMTREX000I RSCS (VER 1, LEV 1, mm/dd/yy) READY
           !
           CP
           logoff
           logon rscs
           ipl 191
DMTREX000I RSCS (VER 1, LEV 1, mm/dd/yy) READY
start newyork
         .
         .
         .
         .
         installation RSCS operation
         .
         .
```

Note: If you are logged on as MAINT, you IPL 195, but if you are logged on as RSCS, you IPL 191 to IPL the RSCS system disk.

Appendixes

A. IBM Programs Executable under CMS

B. Configuration Aid

C. Representative Directory Entries

D. CP/CMS Regeneration Requirements

E. Notational Conventions

F. VM/370 Service Programs

G. Compatibility of VM/370 with CP-67/CMS

H. Creating a 3340 System Residence Volume from a Current 2314 or 3330 System Residence Volume and the PLC Tape

VM/370 ASSEMBLER

A VM/370 Assembler is distributed as a part of the VM/370 system and is required for installation and further support of the system. All necessary installation and support macros are provided in CMS libraries.

The Conversational Monitor System (CMS) is a component of VM/370 and is distributed with it. Certain other facilities mentioned in this publication are not part of VM/370, but can be separately ordered from IBM. These include: IBM System/360 and System/370 operating systems, IBM language processors and other program products, IBM Installed User Programs, and IBM Field Developed Programs. For more information, contact your IBM representative.

PROGRAM PRODUCTS

Figure 29 lists the IBM Program Products that are executable under CMS.

To find the amount of storage required to install a program product, refer to the appropriate program product publication.

INSTALLED USER PROGRAMS

A text-processing program is available: IBM Installed User Program (IUP) SCRIPT/370 (IBM Program No. 5976-PAF). SCRIPT/370 creates formatted output from one or more CMS files, each of which contains text and/or SCRIPT control words. The SCRIPT files are created and modified at a terminal using the CMS Editor.

SCRIPT/370 manuscript facilities include right margin justification, line centering, inserting top and bottom titles, and the ability to invoke additional SCRIPT input files from the file being processed. Other facilities to assist in the preparation of large documents include symbolic capabilities that can generate a table of contents, and number pages and figures.

IBM Program Product	IBM Program Number
OS Code & Go FORTRAN	5734-F01
OS FORTRAN IV (G1)	5734-F02
OS FORTRAN IV Library (Mod I)	5734-LM1
OS FORTRAN IV (H) Extended	5734-F03
OS FORTRAN Library (Mod II)	5734-LM3
FORTRAN Interactive Debug	5734-F05
OS/VS COBOL Compiler and Library	5740-CB1
OS/VS COBOL Library Only	5740-LM1
OS Full American National Standard COBOL Version 4 Compiler and Library	5734-CB2
OS Full American National Standard COBOL Version 4 Library	5734-LM2
OS COBOL Interactive Debug	5734-CB4
OS PL/I Optimizing Compiler	5734-PL1
VS BASIC Processor	5748-XX1
VS BASIC	5734-XX1
OS PL/I Resident Library	5734-LM4
OS PL/I Transient Library	5734-LM5
OS PL/I Optimizing Compiler and Libraries	5734-PL3
OS PL/I Checkout Compiler	5734-PL2
Planning Systems Generator/CMS (PSG/CMS)	5748-XT1

Figure 29. IBM Program Products

A time-sharing facility is available as an IUP: the McGill University System for Interactive Computing (MUSIC), IBM Program No. 5796-AAT. MUSIC is a conversational time-sharing operating system that can execute in a virtual machine under VM/370.

System/370 Model	1401 1440 1460	1401 1440 1410 1460	7070 7074	7080	709 7090 7094 7094II
135	#4457				
145	#4457	#4458			
155 II,158		#3950	#7117		
165 II,168			#7117	#7118	#7119

Figure 30. Integrated Emulators that Execute Under VM/370

INTEGRATED EMULATORS

Emulator-dependent programs (except for DOS emulation under OS or OS/VS) that execute on a particular System/370 equipped with the appropriate compatibility features can execute on that System/370 in DOS or OS virtual machines under VM/370.

Figure 30 shows, by System/370 model number, which integrated emulators can execute under VM/370 and the compatibility feature numbers (#xxxx) that are required.

No changes are required to the emulators, to DOS or OS, or to VM/370 itself to allow emulator-dependent programs to execute in virtual machines.

On the System/370 Model 158 only, the Virtual Machine Assist Feature cannot operate concurrently with the 7070/7074 compatibility feature (Feature #7117).

Appendix B shows the devices and control units that can be specified in a VM/370 system generation; they are grouped by use. It lists the control units, notes the maximum number that can be specified in the FEATURE= operand of the RCTLUNIT macro, and tells whether or not the control units can operate on a shared subchannel.

It shows the devices that can be attached to each control unit, and lists the operands that can be specified for each device in the RDEVICE macro.

The control units and devices are placed in subgroups according to the ways they can be configured. For example, the chart of tape devices indicates that a 2401, 2402, or 2420 can be attached to a 2803 or 2804 control unit.

	RCTLUNIT		Shared Sub-channel	RDEVICE	
Type of Device	CUTYPE=	FEATURE=		DEVTYPE=	Other Operands
Tape Devices	3803	16-DEVICE	yes	3420	MODEL=3, 4, 5, 6, 7 or 8 FEATURE=7-TRACK,DUALDENS
System Consoles	1052	—	—	1052	—
	3210	—	—	3210	—
	3215	—	—	3215	—
	2150	—	—	2150	—
	3066	—	—	3066	—
	3158	—	—	3158	—
Transmission Control Units	2701	—	—	2701	ADAPTER=BSCA, IBM1, or TELE2
	2702	16-DEVICE	—	2702[1]	ADAPTER=BSCA, IBM1, or TELE2 SETADDR=0, 1, 2, or 3
	2703	176-DEVICE	—	2703[1]	ADAPTER=BSCA, IBM1, or TELE2
	3704 3705	16-DEVICE 256-DEVICE	—	3704 3705	ADAPTER=BSCA, IBM1, TELE2, TYPE1, or TYPE2 MODEL=1, 2, 3, 4, 5, 6, 7, or 8 SETADDR=0, 1, 2, or 3 CPTYPE=EP, NCP, or PEP CPNAME=ncpname MAXDIAL=nn BASEADD=cuu

[1] Specify 2702 or 2703 as the device type for CPT-TWX (33/35), 2741, 1050, and 3767 (operating as a 2741) terminals.

Type of Device	RCTLUNIT CUTYPE=	RCTLUNIT FEATURE=	Shared Sub-channel	RDEVICE DEVTYPE=	RDEVICE Other Operands
Transmission Control Units (cont)	ICA	16-DEVICE	---	ICA	ADAPTER=BSCA, IBM1, or TELE2
	2955	---	---	2955	---
Display Devices	2848	32-DEVICE	yes	2260 1052	---
	2845	---	yes	2265	---
	2250	---	---	2250	---
	3272	32-DEVICE	yes	3277 3284 3286	FEATURE=OPRDR
Remote 3270 Display Devices	2701	---	---	2701	ADDRESS=cuu (line address) ADAPTER=BSCA CLUSTER=label
	2703	---	---	2703	ADDRESS=cuu (line address) ADAPTER=BSCA CLUSTER=label
	ICA	---	---	ICA	ADDRESS=cuu (line address) ADAPTER=BSCA CLUSTER=label
	3704 3705	---	---	3704 3705	ADDRESS=cuu (line address) ADAPTER=BSCA CPTYPE=EP BASEADD=cuu CLUSTER=label
Direct Access Storage Devices	2841	---	yes	2311 2321 2303	
	2314 2319 IFA	---	yes	2314 2319	
	3830 3345 ISC	32-DEVICE 16-DEVICE 16-DEVICE	--- --- yes	3330 3333	MODEL=1, 2, or 11 MODEL=1 or 11
	3830 3345 ISC IFA	32-DEVICE 16-DEVICE 64-DEVICE		3340	
	2820	---	yes	2301	

Type of Device	RCTLUNIT CUTYPE=	RCTLUNIT FEATURE=	Shared Sub-channel	RDEVICE DEVTYPE=	RDEVICE Other Operands
Direct Access Storage Devices (cont)	2835	—	—	2305	MODEL=1 or 2
Tape Devices	2803 2804	16-DEVICE 16-DEVICE	yes	2401	MODEL=1, 2, 3, 4, 5, 6, or 8 FEATURE=7-TRACK, DUALDENS
				2402	MODEL=1, 2, 3, 4, 5 or 6 FEATURE=7-TRACK, CONV, DUALDENS
				2420	MODEL=5 or 7
	2403 2404	16-DEVICE	yes	2403 2404	MODEL=1, 2, 3, 4, 5 or 6 FEATURE=7-TRACK, CONV, DUALDENS
	2415	—	yes	2415	MODEL=1, 2, 3, 4, 5 or 6 FEATURE=7-TRACK, CONV
	3411	—	yes	3410 3411	MODEL=1, 2, or 3 FEATURE=7-TRACK, DUALDENS
	3803	16-DEVICE	yes	3420	MODEL=3, 4, 5, 6, 7 or 8 FEATURE=7-TRACK, DUALDENS
Unit Record Output Devices	2821	—	—	1403	CLASS=(class[,class...]) FEATURE=UNVCHSET
				2540P	CLASS=(class[,class...])
	1442 1443	—	—	1442P 1443	 CLASS=(class[,class...])
	3811	—	—	3211	CLASS=(class[,class...])
	2826	—	—	1018	
	2520	—	—	2520P	CLASS=(class,[class...])
	3505	—	—	3525	CLASS=(class,[class...])
Unit Record Input Devices	2821	—	—	2540R	
	2520	—	—	2520R	
	3505	—	—	3505	
	1442	—	—	1442R	
	2495	—	—	2495	
	2822	—	—	2671	

Type of Device	RCTLUNIT CUTYPE=	RCTLUNIT FEATURE=	Shared Sub-channel	RDEVICE DEVTYPE=	RDEVICE Other Operands
Unit Record Input Devices	2826	--	--	1017	
	2501	--	--	2501	
Special Devices	CTCA	--	yes	CTCA	

Appendix C: Representative Directory Entries

The following sample VM/370 directory entries provide an operating system with the entries necessary for operation and updating. The indenting shown in the sample directory entries is for clarity only, and is not required by the Directory program. Blank cards and cards with an asterisk (*) are ignored. LINK control statements are used whenever possible to minimize the number of changes to the VM/370 directory whenever a minidisk extent is moved. A brief explanation of some of the virtual machine userids follows.

AN OPERATOR'S VIRTUAL MACHINE (OPERATOR)

The userid for this virtual machine must be the same as the userid that you specified on the SYSOPER operand of the SYSOPR macro. The operator is given all command privilege classes except class F. Actually, if other virtual machines are defined with command privilege classes appropriate for updating VM/370, the operator's virtual machine only needs class A command privileges. The 191 minidisk that is defined contains CMS files and EXEC procedures, along with the service prcgrams used to update VM/370.

A VIRTUAL MACHINE THAT RECEIVES SYSTEM DUMPS (OPERATNS)

The userid for this virtual machine is the userid you specified on the SYSDUMP operand of the SYSOPR macro. OPERATNS is the default userid. All abnormal termination dumps are sent to this virtual machine. This user is normally given command privilege classes A, B, C, and E. You also need a virtual machine that contains all of the disks normally attached to the system described as full-volume minidisks if you want to rewrite the VM/370 directory by using the DIRECT command. The operations group can examine any disk while it is attached to the system, if you define these disks as full-volume minidisks.

Also, if you plan to use the DDR program to back up the system, you should have all the disks that are normally backed up assigned a fixed virtual address. Then a CMS file containing the DDR control statements can be used to back up the system. Users should not be allowed to write on any disk while that disk is being backed up by the system. You can define additional virtual machines for system backup so that several disks can be backed up at the same time. When you back up several disks at the same time the tape operations can be overlapped.

A VIRTUAL MACHINE FOR UPDATING AND SUPPORTING VM/370 (MAINT)

This virtual machine (MAINT) can support and update the VM/370 system. The 194 minidsk is be used to contain any alterations or fixes to the CP system after they are thcroughly tested and considered stable. The 394 minidisk contains the distributed source files. The MAINT virtual machine has class E and class G command privileges, so that it can issue the SAVESYS command to save CMS. The 190 minidisk contains the CMS nucleus and all of the CMS modules and EXEC procedures available to all

CMS users. Any virtual machine that wants to use the CMS system, links to this disk (190). The 393 minidisk contains the distributed CMS source code in an unaltered form. The 193 minidisk holds CMS PTFs and updates.

A HARDWARE SERVICE VIRTUAL MACHINE (SRMAINT)

This virtual machine (SRMAINT) is normally used by the hardware service representative. Usually, this is the only virtual machine that has command privilege class F. When a virtual machine with class F command privileges logs on, outboard recording is terminated. With class F command privileges, you can set the recording mode for a device, set the mode for recording soft errors, and execute the EREP program. The minidisk is used for diagnostics, CMS EXEC procedures, the CPEREP program, or any other programs used by the hardware support representative.

CMS VIRTUAL MACHINES (USER1, USER2, AND USER3)

These virtual machines have a typical configuration for CMS users; they each have a minimum CMS configuration.

A BATCH VIRTUAL MACHINE (BATCH)

The sample batch virtual machine (BATCH) has a timer and the ISAM option. These may or may not be needed depending upon the type of batch system executing: OS, DOS or CMS. This example also shows that two disks are dedicated to this virtual machine. They are BATCH1 and BATCH2. They are attached to this virtual machine at logon time if they are mounted and not in use by anyone on the system.

A VM/370 VIRTUAL MACHINE (TESTSYS)

This virtual machine has the ECMODE option, and could be used to check a new CP nucleus before moving that nucleus to the flcor system. TESTSYS contains two minidisks, 330 and 331. These disks are exact copies of the real system residence and scratch volumes. They are formatted and allocated so that the real system can spool and page on these disks. If you need additional disks, link to those disks before you IPL the virtual VM/370 system.

THE REMOTE SPOOLING COMMUNICATIONS SUBSYSTEM (RSCS) VIRTUAL MACHINE

This virtual machine executes the Remote Spooling Communications Subsystem component of VM/370. Files to be transmitted are addressed to the virtual reader. Virtual punches and printers are dynamically defined and detached by RSCS, as required. The 190 virtual disk contains the RSCS nucleus, RSCS modules, and the line driver programs. The binary synchronous lines, dedicated to RSCS, are allocated and deallocated by RSCS as required to transmit files to and from remote stations.

The following is a sample VM/370 directory file containing entries that correspond to the preceding virtual machine descriptions.

```
FILE:  USER  DIRECT

*     USER DIRECTORY DECK

DIRECTORY 330 3330 SYSRES

     USER OPERATOR PASSWORD 256K 1M ABCDEG
          ACCOUNT NUMBER BIN1
               CONSOLE 009 3215
               SPCOL C 2540 READER A
               SPOOL D 2540 PUNCH A
               SPOOL E 1403 A
               LINK CMSSYS 190 190 RR
               MDISK 191 2314 1 10 UDISK1 WR RPASS WPASS

     USER OPERATNS PASSWORD 256K 1M ABC
          ACCCUNT NUMBER BIN2
               CONSOLE 009 3215
               SPCOL C 2540 READER A
               SPOOL D 2540 PUNCH A
               SPOOL E 1403 A
               LINK CMSSYS 190 190 RR
               MDISK 191 2314 101 10 UDISK1 WR RPASS WPASS
               MDISK 330 3330 0 404 SYSRES WR RPASS WPASS
               MDISK 331 3330 0 404 SYSWRK RR RPASS WPASS
               MDISK 230 2314 0 203 UDISK1 RR RPASS WPASS
               MDISK 231 2314 0 203 UDISK2 RR RPASS WPASS
               MDISK 232 2314 0 203 BATCH1 RR RPASS WPASS
               MDISK 233 2314 0 203 BATCH2 RR RPASS WPASS
               MDISK 234 2314 0 203 APLRES RR RPASS WPASS
               MDISK 235 2314 0 203 APLWRK RR RPASS WPASS

     USER MAINT CPCMS 512K 16M BCEG
          ACCCUNT installation defined
               OPTION ECMODE REALTIMER
               CONSOLE   009   3215
               SPOOL     00C   2540   READER A
               SPOOL     00D   2540   PUNCH  A
               SPOOL     00E   1403   A
               MDISK 190 2314 035 110 CPV2L0 MR READ
               MDISK 191 2314 019 010 CPV2L0 MR READ
               MDISK 194 2314 145 058 CPV2L0 MR READ
               MDISK 199 2314 034 001 CPV2L0 WR READ
               MDISK 193 2314 001 050 USERD1 MR READ
               MDISK 294 2314 051 050 USERD1 MR READ
               MDISK 393 2314 001 110 USERD2 MR READ
               MDISK 394 2314 001 110 USERD3 MR READ
               MDISK 390 2314 101 003 USERD1 MW READ
               MDISK xxx 2314 000 203 yyyyyy MW

     USER SRMAINT PASSWORD 256K 1M FG
          ACCOUNT NUMBER BIN5
               CONSOLE 009 3215
               SPOOL C 2540 READER A
               SPOOL D 2540 PUNCH A
               SPOOL E 1403 A
               LINK CMSSYS 190 190 RR
               MDISK 191 2314 51 10 UDISK1 WR RPASS WPASS
```

```
USER USER1 PASSWORD
     ACCOUNT NUMBER BIN7
          CONSOLE 009 3215
          SPOOL C 2540 READER
          SPOOL D 2540 PUNCH
          SPOOL E 1403
          LINK CMSSYS 190 190 RR
          MDISK 191 2314 71 10 UDISK1 WR RPASS WPASS

USER USER2 PASSWORD
     ACCOUNT NUMBER BIN8
          CONSOLE 009 3215
          SPOOL C 2540 READER
          SPOOL D 2540 PUNCH
          SPOOL E 1403
          LINK CMSSYS 190 190 RR
          MDISK 191 2314 81 10 UDISK1 WR RPASS WPASS

USER USER3 PASSWORD
     ACCCUNT NUMBER BIN9
          CONSOLE 009 3215
          SPOOL C 2540 READER
          SPOOL D 2540 PUNCH
          SPOOL E 1403
          LINK CMSSYS 190 190 RR
          MDISK 191 2314 91 10 UDISK1 WR RPASS WPASS

USER BATCH PASSWORD 512K
     ACCCUNT NUMBER BIN10
     OPTION REALTIMER ISAM
          CONSOLE 009 3215
          SPOOL C 2540 READER
          SPOOL D 2540 PUNCH
          SPOOL E 1403
          DEDICATE 230 BATCH1
          DEDICATE 231 BATCH2

USER TESTSYS PASSWORD 512K
     ACCOUNT NUMBER BIN11
     OPTION ECMODE
          CONSOLE 01F 3215
          SPOOL C 2540 READER
          SPOOL D 2540 PUNCH
          SPOOL E 1403
          LINK CMSSYS 190 190 R
          MDISK 330 3330 1 15 SYSWRK WR RPASS WPASS
          MDISK 331 3330 16 20 SYSWRK WR RPASS WPASS

USER RSCS PASSWORD 512K
     ACCOUNT NUMBER BIN11A
          CONSOLE 01F 3215
          SPOOL C 2540 READER
          MDISK 190 2314 020 005 CMSVL1 R
          DEDICATE 0B1 078
          DEDICATE 0B2 079
          DEDICATE 0B3 07A

USER 2780 PASSWORD
     ACCOUNT NUMBER BIN12
     OPTION REALTIMER
          CONSOLE 01F 3215
          SPOOL C 2540 READER
          SPOOL D 2540 PUNCH
          SPOOL E 1403
          DEDICATE 30 30
```

```
USER JFK PASSWORD 512K
     ACCOUNT NUMBER BIN13
          CONSOLE 01F 3215
          SPOOL C 2540 READER
          SPOOL D 2540 PUNCH
          SPOOL E 1403
          MDISK 230 2314 000 050 OSDOS1 WR
          MDISK 231 2314 100 020 OSDOS1 WR
          MDISK 232 2314 031 030 CPVOL1 WR

USER JCE PASSWORD
     ACCOUNT NUMBER BIN14
          CONSOLE 01F 3215
          SPOOL C 2540 READER
          SPOOL D 2540 PUNCH
          SPOOL E 1403
          MDISK 130 2314 050 050 OSDOS1 WR
          MDISK 131 2314 001 020 CPVOL1 WR

USER APLSYS PASSWORD
     ACCOUNT NUMBER BIN15
          CONSOLE 01F 3215
          SPOOL C 2540 READER
          SPOOL D 2540 PUNCH
          SPOOL E 1403
          DEDICATE 191 APLSYS
          DEDICATE 190 APLWRK
          SPECIAL 80 2702 IBM
          SPECIAL 81 2702 IBM
          SPECIAL 82 2702 IBM
          SPECIAL 83 2702 IBM

USER OS2 PASSWORD
     ACCOUNT NUMBER BIN16
          CONSOLE 01F 3215
          SPOOL C 2540 READER
          SPOOL D 2540 PUNCH
          SPOOL E 1403
          LINK JFK 230 230 RR
          LINK CMSSYS 190 190 RR
          MDISK 231 2314 120 82 W
          MDISK 191 2314 101 10 UDISK1 WR RPASS WPASS
```

Appendix D: CP/CMS Nucleus/Module Regeneration Requirements

Whenever a PTF is applied to CMS source code, either the CMS nucleus, or some CMS module, or both, must be regenerated. The following table shows which must be regenerated in each case. (If a source name does not appear in the table, only the CMS nucleus needs to be regenerated.)

Change in Source	Requires Regeneration of Module/Nucleus	EXEC Procedure To Use
DMSACC	ACCESS, Nucleus	CMSGEND
DMSACF	ACCESS, Nucleus	CMSGEND
DMSACM	ACCESS, Nucleus	CMSGEND
DMSALU	ACCESS, FORMAT, RELEASE, Nucleus	CMSGEND
DMSARE	RELEASE	CMSGEND
DMSASD	ASSEMBLE	ASMGEND
DMSASM	ASSEMBLE	CMSGEND
DMSBSC	BASIC	BSCGEND
DMSBTB	CMSBATCH	CMSGEND
DMSCMP	COMPARE	CMSGEND
DMSCPY	COPYFILE	CMSGEND
DMSDSK	DISK	CMSGEND
DMSEDC	EDIT (see Note)	CMSGEND
DMSEDF	EDIT (see Note)	CMSGEND
DMSEDI	EDIT (see Note)	CMSGEND
DMSEDX	EDIT (see Note)	CMSGEND
DMSEXT	DMSEXT	CMSGEND
DMSFLD	FILEDEF	CMSGEND
DMSFOR	FORMAT	CMSGEND
DMSGIO	EDIT (see Note)	CMSGEND
DMSGLB	GLOBAL	CMSGEND
DMSGND	GENDIRT	CMSGEND
DMSHDI	HNDINT	CMSGEND
DMSHDS	HNDSVC	CMSGEND
DMSLBM	MACLIB	CMSGEND
DMSLBT	TXTLIB	CMSGEND
DMSLDS	LISTDS	CMSGEND
DMSLST	LISTFILE	CMSGEND
DMSMDP	MODMAP	CMSGEND
DMSMVE	MOVEFILE	CMSGEND
DMSOVR	SVCTRACE	CMSGEND
DMSOVS	DMSOVS	CMSGEND
DMSPRT	PRINT	CMSGEND
DMSPUN	PUNCH	CMSGEND
DMSQRY	QUERY	CMSGEND
DMSRDC	READCARD	CMSGEND
DMSRNM	RENAME	CMSGEND
DMSSCR	EDIT (see Note)	CMSGEND
DMSSET	SET	CMSGEND
DMSSRT	SORT	CMSGEND
DMSSVT	DMSSVT	CMSGEND
DMSSYN	SYNONYM	CMSGEND
DMSTPD	TAPPDS	CMSGEND
DMSTPE	TAPE	CMSGEND
DMSTYP	TYPE	CMSGEND
DMSUPD	UPDATE	CMSGEND
DMSZAP	ZAP	CMSGEND

Note: When the CMSGEND EXEC procedure is invoked for EDIT, it creates the EDIT module, and then automatically reinvokes itself to create the EDINIT module.

If you must regenerate the CMS nucleus, see the "Generating the CMS Nucleus" and "Using VMFLOAD to Generate a New Nucleus" sections of "Part 6: Updating VM/370."

If a CMS module must be regenerated, see the "Using the CMSGEND EXEC Procedure to Generate A CMS Module" and "Creating CMS Disk-Resident Modules" sections of "Part 6: Updating VM/370."

If you apply a PTF to certain CP source programs, the corresponding CMS modules must be regenerated also to run properly. The source name, module name, and procedures used for regenerating the module are shown in the following table.

Change in Source	Regenerate Module Name Required	EXEC Procedures to Use
DMKDDR	DDR	GENERATE, CMSGEND
DMKDIR	DIRECT	GENERATE, CMSGEND
DMKEDM	VMFDUMP	CMSGEND
DMKFMT	No module name—does not execute under CMS	GENERATE
DMKRND	NCPDUMP	CMSGEND
DMSARN	ASM3705	LOADCC
DMSGRN	GEN3705	CMSGEND
DMSNCP	SAVENCP	CMSGEND

The LOADCC EXEC procedure is described in "Part 4: Generating the 3704/3705 Control Program;" all the other EXEC procedures are described in "Part 6: Updating VM/370."

The notation used to define the command syntax in this publication is:

- Truncations and Abbreviations of Commands

  Where truncation of a command name is permitted, the shortest acceptable version of the command is represented by uppercase letters. (Remember, however, that VM/370 commands can be entered with any combination of upper and lowercase letters.) The example below shows the format specification for the FILEDEF command.

  ```
  FIledef
  ```

  This representation means that FI, FIL, FILE, FILED, FILEDE, and FILEDEF are all valid specifications for this command name.

  Operands and options are specified in the same manner. Where truncation is permitted, the shortest acceptable version of the operand or option is represented by uppercase letters in the command format box. If no minimum truncation is noted, the entire word (represented by all capital letters) must be entered.

  Abbreviations are shorter forms of command names, operands, and options. Abbreviations for command names are shown below the full name in the format box. Abbreviations for operands and options are shown in the description of the individual operands and options that follows the format box. For example, the operand READER has both a minimum truncation and an abbreviation. In the format box it is shown as:

  ```
  Reader
  ```

  indicating that the minimum truncation is R. In the discussion of the READER operand that follows, it is shown as:

  ```
  READER
  RDR
  ```

  indicating that the abbreviation is RDR. Thus, the acceptable specifications for the READER operand are: R, RE, REA, READ, READE, READER, and RDR.

  In some cases what appears to be a minimum truncation is really the only valid abbreviation. For example, the abbreviation for MEMBER is MEM. Only these two forms are valid and no truncations are allowed. The format box contains

  ```
  MEMBER {name}
         { *  }
  ```

  and the description that follows the format box is

  ```
  MEMBER {name}
  MEM    { *  }
  ```

- The following symbols are used to define the command format and should never be typed when the actual command is entered.

underscore	_
braces	{ }
brackets	[ ]
ellipsis	...

- Uppercase letters and words, and the following symbols, should be entered as specified in the format box.

asterisk	*
comma	,
hyphen	-
equal sign	=
parentheses	( )
period	.
colon	:

- Lowercase letters, words, and symbols that appear in the command format box represent variables for which specific informaticn should be substituted. For example, "fn ft fm" indicates that file identifiers such as "MYFILE EXEC A1" should be entered.

- Choices are represented in the command format boxes by stacking.

  A
  B
  C

- An underscore indicates an assumed default option. If an underscored choice is selected, it need not be specified when the command is entered.

  Example
  The representation

  A
  B
  C

  indicates that either A, B, or C may be selected. However, if B is selected, it need not be specified. Or, if none is entered, B is assumed.

- The use of braces denotes choices, one of which must be selected.

  Example
  The representation

  indicates that you must specify either A, or B, or C. If a list of choices is enclosed by neither brackets or braces, it is to be treated as if enclosed by braces.

- The use of brackets denotes choices, one of which may be selected.

  Example
  The representation

  indicates that you may enter A, B, or C, or you may omit the field.

- An ellipsis indicates that the preceding item or group of items may be repeated more than once in succession.

  Example
  The representation

```
(options...)
```

  indicates that more than one option may be coded within the parentheses.

Appendix F: VM/370 Service Programs Used During System Generation

The following VM/370 service programs are used when you generate VM/370:

- The VM/370 Directory program (DMKDIR)
- The DASD Dump Restore program (DMKDDR)
- The IBCDASDI Virtual Disk Initialization program (IBCDASDI)
- The Format/Allocate program (DMKFMT)

The VM/370 Directory program is described in "Part 2: Defining Your VM/370 System"; the others are described in this Appendix.

DASD DUMP RESTORE (DDR) SERVICE PROGRAM AND HOW TO USE IT

Use the DASD Dump Restore (DDR) service program to dump, restore, copy, or print VM/370 user minidisks. The DDR program may run as a standalone program, or under CMS via the DDR command.

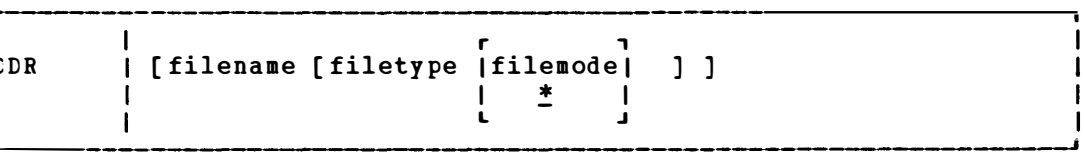
The DDR program has five functions:

1. Dumps part or all of the data from a DASD device to tape.
2. Transfers data from tapes created by the DDR DUMP function to a direct access device. The direct access device must be the same as that which originally contained the data.
3. Copies data from one device to another of the same type. Data may be reordered, by cylinder, when copied from disk to disk. In order to copy one tape to another, the original tape must have been created by the DDR DUMP function.
4. Prints selected parts of DASD and tape records in hexadecimal and EBCDIC on the virtual printer.
5. Displays selected parts of DASD and tape records in hexadecimal and EBCDIC on the terminal.

To generate the VM/370 starter system from the distribution tape, the standalone RESTORE function must be used.

INVOKING DDR UNDER CMS

The format of the DDR command is:



where:

filename filetype [filemode] is the identification of the file containing the control statements for the DDR program. If no file identification is provided, the DDR program attempts to obtain control statements from the console. The filemode defaults to * if a value is not provided.

Note: If you use the CMS DDR command, CMS ignores the SYSPRINT control statement and directs the output to the CMS printer 00E.

INVOKING DDR AS A STANDALONE PROGRAM

To use DDR as a standalone program, the operator should IPL it from a real or virtual IPL device as he would any other standalone program. Then indicate where the DDR program is to obtain its control statements by responding to prompting messages at the console.

DDR CONTROL STATEMENTS

DDR control statements describe the intended processing and the needed I/O devices. I/O definition statements must be specified first.

All control statements may be entered from either the console or the card reader. Only columns 1 to 71 are inspected by the program. All data after the last operand in a statement is ignored. An output tape must have the DASD cylinder header records in ascending sequences; therefore, the extents must be entered in sequence by cylinder. Only one type of function - dump, restore, or copy - may be performed in one execution, but up to 20 statements describing cylinder extents may be entered. The function statements are delimited by an input or output statement, or by a null line if the console is used for input. If additional functions are to be performed, the sequence of certain control cards must be repeated. Only those statements needed to redefine the I/O devices are necessary for subsequent steps. All other I/O definitions remain the same.

To return to CMS, enter a null line (carriage return) in response to the prompting message (ENTER:). To return directly to CP, key in #CP.

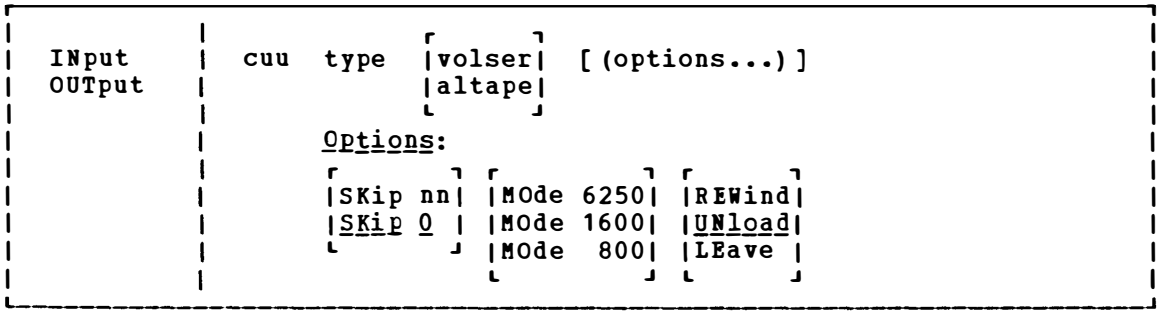
The PRINT and TYPE statements work differently from other DDR control statements in that they operate on only one data extent at a time. If the input is from a tape created by the dump function, it must be positioned at the header record for each step. The PRINT and TYPE statements have an implied output of either the console (TYPE) or system printer (PRINT). Therefore, PRINT and TYPE statements need not be delimited by an input or output statement.

I/O DEFINITION STATEMENTS

The I/O definition statements describe the tape, DASD, and printer devices used while executing the DASD Dump Restore program.

INPUT/OUTPUT Control Statement

An INPUT or OUTPUT statement describes each tape and DASD unit used. The format of the INPUT/OUTPUT statement is:



where:

INPUT indicates that the device described is an input device.

OUTPUT indicates that the device described is an output device.

cuu is the unit address of the device.

type is the device type (2314, 2319, 3330, 3330-11, 3340-35, 3340-70, 2305-1, 2305-2, 2400, 2420, or 3420) (no 7-track support for any tape devices). Specify a 3410 device as a 3420, a 3340-70F as a 3340-70, and a 3333 as a 3330.

Note: The DASD Dump Restore (DDR) program, executing in a virtual machine, uses I/O DIAGNOSE 20 to perform I/O operations on tape and DASD devices. DDR under CMS requires that the device type entered agree with the device type of the real device as recognized by VM/370. If there is a conflict with device types, the following message is issued:

```
DMKDDR701E INVALID OPTION
```

However, if DDR executes standalone in a virtual machine, DDR uses DIAGNOSE 20 to perform the I/O operation if the device types agree and SIO if the device types do not agree.

volser is the volume serial number of a DASD device. If the keyword 'SCRATCH' is specified instead of the volume serial number, no label verification is performed.

altape is the address of an alternate tape drive.

Note: If multiple reels of tape are required and "altape" is not specified, DDR types the following at the end of the reel:

```
END OF VOLUME CYL xxx HD xxx, MOUNT NEXT TAPE
```

After the new tape is mounted, DDR continues automatically.

Options

SKIP nn
0 forward spaces nn files on the tape. nn is any number up to 255. The SKIP option is reset to zero after the tape has been positioned.

MODE 6250
MODE 1600
MODE 800 causes all output tapes that are opened for the first time and at the load point to be written or read in the specified density. All subsequent tapes mounted are also set to the specified density. If no mode option is specified, then no mode set is performed and the density setting remains as it previously was.

REWIND rewinds the tape at the end of a function.

UNLOAD rewinds and unloads the tape at the end of a function.

LEAVE leaves the tape positioned at the end of the file at the end of a function.

SYSPRINT Control Statement

Use the SYSPRINT control statement (in the standalone application only) to describe the printer that is to print data extents specified by the PRINT statement. It also can print a map of the cylinder extents from the DUMP, RESTORE, or COPY statement. If the SYSPRINT statement is not provided, the printer assignment defaults to 00E. CMS ignores the SYSPRINT statement when you invoke DDR as a command under CMS, and CMS always directs the output to 00E. The format of the SYSPRINT control statement is:

```
SYsprint | cuu
```

where:

cuu specifies the unit address of the device.

Function Statements

The function statements tell the DDR program what action to perform. The function commands also describe the extents to be dumped, copied, or restored. The format of the DUMP/COPY/RESTORE control statement is:

```
          | ┌                                                ┐
DUmp      | |cyl1 [To] [cyl2 [Reorder] [To] [cyl3]]          |
COpy      | |CPvol                                           |
REstore   | |ALL                                             |
          | |NUcleus                                         |
          | └                                                ┘
```

where:

DUMP requests the program to move data from a direct access volume onto a magnetic tape or tapes. The data is moved cylinder by cylinder. Any number of cylinders may be moved. The format of the resulting tape is:

Record 1: a volume header record, consisting of data describing the volumes.

Record 2: a track header record, consisting of a list of count fields to restore the track, and the number of data records written on tape. After the last count field the record contains key and data records to fill the 4K buffer.

Record 3: track data records, consisting of key and data records packed into 4K blocks, with the last record truncated.

Record 4: either the end-of-volume or end-of-job trailer label. The end volume label contains the same information as the next volume header record except that the ID field contains EOV. The end-of-job trailer label contains the same information as record 1 except that the cylinder number field contains the disk address of the last record on tape and the ID field contains EOJ.

COPY requests the program to copy data from one device to another device of the same or equivalent type. Data may be recorded on a cylinder basis from input device to output device. A tape-to-tape copy can be accomplished only with data dumped by this program.

RESTORE requests the program to return data that has been dumped by this program. Data can be restored only to a DASD volume of the same or equivalent device type from which it was dumped. It is possible to dump from a real disk and restore to a minidisk as long as the device types are the same.

cyl1 [TO] [cyl2 [REORDER] [TO] [cyl3]]
Only those cylinders specified are moved, starting with the first track of the first cylinder (cyl1), and ending with the last track of the second cylinder (cyl2). The REORDER operand causes the output to be reordered, starting at the specified cylinder (cyl3) or at the starting cylinder (cyl1) if "cyl3" is not specified. The REORDER operand must not be specified unless specified limits are defined for the operation; the starting and, if required, ending cylinders (cyl1 and cyl2) must be specified.

CPVOL specifies that cylinder 0 and all active directory and permanent disk space are to be copied, dumped, or restored. This indicates that both source and target disk must be in CP format, that is, the CP Format/Allocate service program must have formatted them.

ALL specifies that the operation is to be performed on all cylinders.

NUCLEUS specifies that record 2 on cylinder 0, track 0 and the nucleus cylinders will be dumped, copied, or restored.

Restrictions:

1. Each track must contain a valid home address, containing the real cylinder and track location.

2. Record zero must not contain more than eight key and/or data characters.

3. Flagged tracks are treated just as any other track for all 2314, 2319, 3340, and 2305 devices. That is, no attempt is made to substitute the alternate track data when a defective primary track is read. In addition, tracks are not inspected to determine whether they were previously flagged when written. Therefore, volumes containing flagged tracks should be restored to the same cylinders of the volume from which they were dumped. The message DMKDDR715E occurs each time a defective track is dumped, copied or restored, and the operation continues.

4. Flagged tracks for a 3330 device are handled automatically by the control unit and may never be detected by the program. The program may detect a flagged track if, for example, no alternate track is assigned to the defective primary track. If a flagged track is detected by the program, message DMKDDR715E occurs and the operation terminates.

Example:

```
INPUT 191 3330 SYSRES
OUTPUT 180 2400 181 (MODE 800
SYSPRINT 00F
DUMP CPVOL
INPUT 130 3330 MINI01
DUMP 1 TO 50 REORDER 51
60 70 101
```

This example sets the density to 800 bpi, then dumps all pertinent data from the volume labeled 'SYSRES' onto the tape that is mounted on unit 180. If the program runs out of room on the first tape, it continues dumping onto the alternate device (181). A map of the dumped cylinders is printed on unit 00F while the program is dumping. When the first function is complete, the volume labeled 'MINI01' is dumped cntc a new tape. Its cylinder header records are labeled 51 to 100. A map of the dumped cylinders is printed on unit 00F. Next, cylinders 60 to 70 are dumped and labeled 101 to 111. This extent is added to the cylinder map on unit 00F. When the DDR processing is complete, the tapes are unloaded and the program stops.

If cylinder extents are being defined from the console, the following is displayed:

```
ENTER CYLINDER EXTENTS
ENTER:
```

For any extent after the first extent, the message

```
ENTER NEXT EXTENT OR NULL LINE
ENTER:
```

is displayed.

The user may then enter additional extents to be dumped, restored, or copied. A null line causes the job step to start.

PRINT/TYPE Function Statement

Use the PRINT and TYPE function statement to print or type (display) a hexadecimal and EBCDIC translation of each record specified. The input device must be defined as direct access or tape. The output is directed to the system console for the TYPE function, or to the SYSPRINT device for the PRINT function. (This does not cause redefinition of the output unit definition.) The format of the PRINT/TYPE control statement is:

PRint TYpe	cyl1 [hh1 [rr1]] [To cyl2 [hh2 [rr2]]] [(options)]
	options: [Hex] [Graphic] [Count]

where:

cyl1 is the starting cylinder.

hh1 is the starting track. If present, it must follow the cyl1 operand. The default is track zero.

rr1 is the starting record. If present, it must follow the hh1 operand. The default is home address and record zero.

[TO] cyl2 is the ending cylinder. If more than one cylinder is to be printed or typed "TO cyl2" must be specified.

hh2 is the ending track. If present, it must follow the cyl2 operand. The default is the last track on the ending cylinder.

rr2 is the record ID of the last record to print. The default is the last record on the ending track.

Options:

HEX prints or displays a hexadecimal representation of each record specified.

GRAPHIC prints or displays an EBCDIC translation of each record specified.

COUNT prints or displays only the count field for each record specified.

Examples:

```
PRINT 0 TO 3
```

Prints all of the records from cylinders 0, 1, 2, and 3.

```
PRINT 0 1 3
```

Prints only one record, from cylinder 0, track 1, record 3.

```
PRINT 1 10 3 TO 1 15 4
```

Prints all records starting with cylinder 1, track 10, record 3, and ending with cylinder 1, track 15, record 4.

The example in Figure 31 shows the information displayed at the console (TYPE function) or system printer (PRINT function) by the DDR program. The listing is annotated to describe some of the data fields.

Responses

DMKDDR725R ORIGINAL INPUT DEVICE WAS(IS) LARGER THAN OUTPUT DEVICE. DO YOU WISH TO CONTINUE? RESPONSE YES OR NO:

Explanation:
RESTORE function - The number of cylinders on the original DASD input unit is compared with the number of cylinders on the output device.

COPY function - The input device contains more cylinders than the output device.

Operator Action: The operator must determine if the COPY or RESTORE function is to continue. The response is either yes or no.

DMKDDR711R VOLID READ IS volid2 [NOT volid1]
DO YOU WISH TO CONTINUE? RESPOND YES NO OR REREAD:

where:

volid2 is the volume serial number from the VOL1 label on the DASD unit.

volid1 is the volume serial number from the INPUT or OUTPUT control card.

The volume serial number read from the device at cuu is not the same as that specified on the INPUT or OUTPUT control card.

DMKDDR716R NO VOL1 LABEL FOUND FOR volser
DO YOU WISH TO CONTINUE? RESPOND YES NO OR REREAD:

where:

volser is the volume serial number of the DASD device from the INPUT or the OUTPUT control card.

The DASD device at cuu contains no volume serial number.

DMKDDR717R DATA DUMPED FROM volid1 TO BE RESTORED to volid2
DO YOU WISH TO CONTINUE? RESPOND YES NO OR REREAD:

where:

volid1 is the volume serial number from the input tape header record (volume dumped).

volid2 is the volume serial number from the output DASD device.

The above message is printed to verify the input parameters.

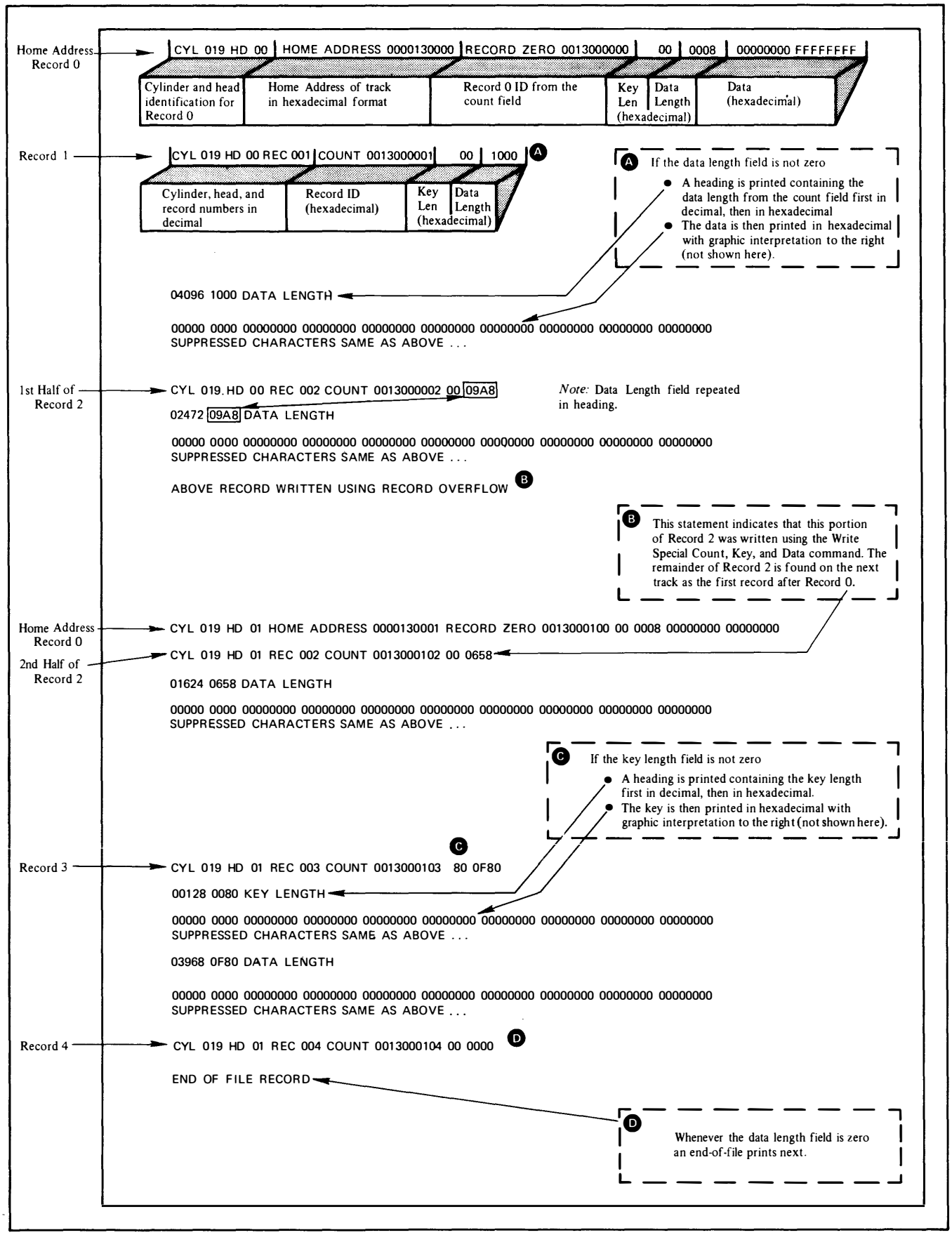


Figure 31. Annotated Sample of Output from the TYPE and PRINT Functions of the DDR Program

DISK BACKUP VIA DASD DUMP RESTORE SERVICE PROGRAM (DMKDDR)

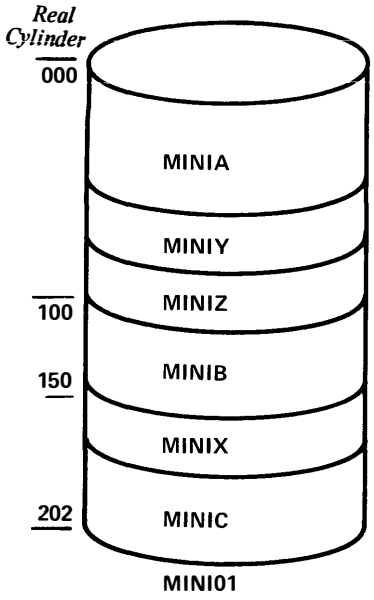
EXAMPLES

Example 1:

You can back up a system volume (MINI01) containing many minidisks and then restore the minidisk named MINIB to a different area on a different disk (MINI05):

To dump the minidisks to tape:

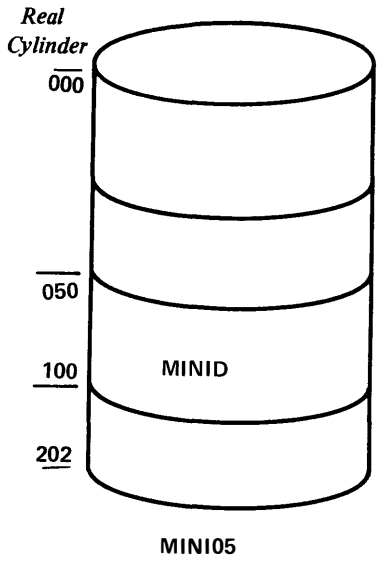
```
INPUT 230 2314 MINI01
OUTPUT 180 3420
DUMP ALL
```



To restore MINIB to the user's MINID minidisk using the tape created above, log on with the userid that owns MINID.

```
INPUT 180 3420
OUTPUT 191 2314 MINID
RESTORE 100 TO 150 REORDER TO 0
```

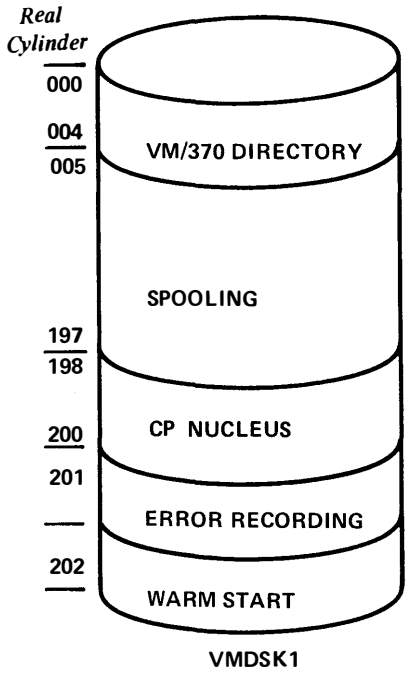
(*Note:* The area allocated to MINID on DASD volume MINI05 would have been defined in the directory. REORDER TO 0 specifies that MINID's virtual cylinders will be 0 to 50.)



| Example 2:

| You can dump the VM/370 system residence volume:

```
INPUT 330 2314 VMDSK1
OUTPUT 180 3420
DUMP CPVOL
```



| When CPVOL is specified with the DUMP or COPY command, cylinder 0 and
| all space allocated for directory and permanent space are dumped or
| copied (DRCT and PERM cylinders are indicated in the byte allocation
| record on cylinder 0). In this example, the spooling space and all
| nonallocated directory space are not dumped or copied.

| When restoring or copying to a VM/370 disk, it is your responsibility to
| ensure that all data not restored or copied is in the proper format as
| described in the byte allocation map of the receiving volume.

| Example 3:

| You can copy real volume VMDSK1 cylinders 100 to 150 to real volume
| VMDSK2 cylinders 50 to 100:

```
|      INPUT  130 2314 VMDSK1
|      OUTPUT 131 2314 VMDSK2
|      COPY   100 TO 150 REORDER TO 50
```

| Example 4:

| You can copy minidisk 190 containing the CMS system to minidisk 199:

```
|      INPUT  190 2314 CMSSYS
|      OUTPUT 199 2314 CMSSYS
|      COPY   0 TO 55
```

| (The copied CMS system does not require any modifications to be used as
| a system disk.)

IBCDASDI -- GENERAL INFORMATION

VM/370 does virtual disk initialization with the OS utility program IBCDASDI. IBCDASDI formats real or virtual VM/370 disk volumes for OS and DOS use. Do not use it to format CP disk areas (for paging, spooling, and so forth), or CMS disk areas. The execution of IBCDASDI is performed from the virtual card reader and the technique is described in "Invoking IBCDASDI" later in this section.

INITIALIZING A MINIDISK

IBCDASDI can, in addition to initializing real disks, initialize a minidisk. A minidisk can be initialized with or without a surface analysis (a test for defective tracks); a surface analysis should be included when a minidisk is initialized for the first time. Tracks in the last cylinders of a 2314 minidisk are left open for assignment as alternate tracks. No tracks are saved for alternates on 3330 or 3340 minidisks.

IBCDASDI Restrictions:

The IBCDASDI program cannot check to see if the 3330 or 3340 space to be initialized was previously formatted.

If you format only 5 cylinders of a 5 cylinder virtual disk of a 3330 or 3340 but specify 20 cylinders to be initialized (CYLNO=20), the IBCDASDI program does not initialize all 20 cylinders but merely updates the format 5 label in the VTOC to indicate 20 cylinders without checking for the existence of 20 cylinders and without issuing an error message. Later, any attempt to use the sixth through the twentieth cylinder causes a Seek Check and the channel program abnormally terminates.

Initialization with Surface Analysis:

The IBCDASDI program:

- Checks for tracks that were previously designated as defective (flagged) and have had alternates assigned. The program automatically assigns alternate tracks for 2314/2319 disk devices but not 3330s. This test must be suppressed when a disk is being initialized with surface analysis for the first time. This test must not be suppressed when a disk is initialized without surface analysis.

- Performs a surface analysis of each track of a 2314 or 2319 and automatically assigns alternates (for 2314/2319s), if necessary. Tracks that are available for use as alternates are checked first.

- Writes a track descriptor record (record 0), and erases the remainder of each track. When initializing a disk with surface analysis, IBCDASDI also writes a standard home address.

- Writes IPL records on track 0 (records 1 and 2).

- Writes volume label on track 0 (record 3) and provides space for additional records, if requested.

- Constructs and writes a volume table of contents (VTOC).

- Writes IPL program, if requested, on track 0 (2314, 2305, 2319, 3330 or 3340), or track 1 (2311).

Initialization without Surface Analysis:

The IBCDASDI program:

- Checks for tracks that were previously designated as defective (flagged) and have had alternates assigned. The program automatically assigns alternates (2314/2319 disk devices only). This test must not be suppressed.
- Writes a standard home address, a track descriptor record (record 0), and erases the remainder of each track.
- Writes IPL records on track 0 (records 1 and 2).
- Writes volume label on track 0 (record 3) and provides space for additional records, if requested.
- Constructs and writes a volume table of contents (VTOC).
- Writes IPL program, if requested, on track 0 (2314, 2305, 3330 or 3340 devices) or track 1 (2311 disks).

Note: The IBCDASDI program cannot assign alternate tracks for 3330/3340 volumes, except when specified by the GETALT statement. Even with the GETALT statement, the IBCDASDI program cannot assign alternate tracks for a 3330/3340 minidisk because no cylinder has been allocated on which to assign alternate tracks. Defective tracks are flagged and alternate tracks are assigned when the 3330/3340 storage volumes are initialized at the factory. An IBCDASDI job that initializes a 3330/3340 performs the "Quick DASDI" function, which reads alternate tracks, decrementing the total unit of alternates by one whenever an alternate is found defective or assigned, writes a volume label and VTOC, and writes an IPLTEXT if requested. No surface analysis is performed and no home address or record 0 is written on the primary tracks. The BYPASS and FLAGTEST options of the DADEF statement are ignored.

DASD 3340 disk packs are factory-shipped without flagged tracks and alternate track assignments. IBCDASDI's "Quick DASDI" detects 3340 customer-generated alternate track assignments.

The IBCDASDI program requires the following function-defining statements to initialize direct access volumes; they must appear in the following sequence:

1. JOB Statement; indicates the beginning of the IBCDASDI job.
2. MSG Statement; defines the output device for operator messages.
3. DADEF Statement; defines the DASD device to be initialized.
4. VLD Statement; labels the volume and allocates space for additional labels.
5. VTOCD Statement; controls the location of the volume-table-of-contents (VTOC).
6. IPLTXT Statement (optional); separates service program control statements from IPL text statements.
7. END Statement; indicates the end of an IBCDASDI job or IPL text.
8. LASTCARD Statement (optional); end of a series of stacked IBCDASDI jobs.

JOB Statement

The JOB statement indicates the beginning of a job. The format of the JOB statement is:

```
[name]    | JOB                |[user information]
```

MSG Statement

The MSG statement defines an output device for operator messages. It follows the JCB statement and precedes any function-defining statements that are associated with the IBCDASDI program. The format of the MSG statement is:

```
[name]    | MSG      |TODEV=xxxx,TOADDR=cuu
```

where:

TODEV=xxxx — is the device type of the message output device (1403, 1052, 2400, 3400, 3210, or 3211).

TOADDR=cuu — is the channel number (c) and unit number (uu) of the message output device.

DADEF Statement

The DADEF statement defines the direct access volume to be initialized. The format of the DADEF statement is:

```
[name]     |DADEF    |TODEV=xxxxxx                                    1
           |         |,TOADDR=cuu
           |         |[,IPL=YES]
           |         |{,VOLID=serial }
           |         |{,VOLID=SCRATCH}
           |         |[,MODEL=n]
           |         |[,CYLNO=nnn]
           |         |-----------------------------------------------
           |         |[,FLAGTEST=NO]                                  2
           |         |[,PASSES=n]
           |         |-----------------------------------------------
           |         |[,BYPASS=YES]                                   3
           |         |
-----------------------------------------------------------------------
1Applies to initialization with or without surface analysis.
2Applies to initialization with surface analysis.
3Applies to initialization without surface analysis.
```

Where:

TODEV=xxxxxx	is a four- to six-character device type of the direct access device. Specify either: 2305 2314 3340 3330 for 3330 Models 1 or 2 3330-1 for 3330 Model 11
TOADDR=cuu	is the channel (c) and unit address (uu) of the device. Specify the real address of the device if you are executing on the real machine; specify the virtual address of the device if you are executing on the virtual machine.
IPL=YES	writes an IPL program on the minidisk. An IPL initialization program must be written on a device to be used for system residence. If IPL is omitted, no IPL program is written.
VOLID=serial	is the volume serial number of the minidisk to be initialized. If "serial" matches the volume serial number found on the minidisk to be initialized, the operation proceeds. If it does not match, the operator is notified.
VOLID=SCRATCH	specifies that no volume serial number check is to be made.
MODEL=n	is a decimal model number (1 or 2). This operand is only for the 2305 and corresponds to the 2305-1 and 2305-2, respectively.
CYLNO=nnn	is a decimal number which specifies the number of cylinders to be formatted. If the CYLNO parameter is omitted, IBCDASDI intializes the entire real volume specified.
FLAGTEST=NO	(applies to surface analysis) specifies that the program is not to check for previously flagged tracks before surface analysis is attempted on this device. (FLAGTEST=NO applies only to 2314 and 2319 devices, and should be specified when the disk recording surface is initialized for the first time.) Note: Because no check is ever made for previously flagged tracks on drum volumes, FLAGTEST=NO is not coded when these devices are initialized.
PASSES=n	specifies the number of passes per track to be made in checking for defective tracks. PASSES is valid when surface analysis is to be performed or when a "Quick DASDI" is to be performed on a 3330, 3330-1, or 3340 volume. The value n can be 0 through 255. The 0 specification indicates that a "Quick DASDI" is to be performed on a 3330, 3330-1, or 3340 volume. For a 3330, 3330-1, or 3340 volume, a value other than zero causes record 0 to be written on each track. No check is made for defective tracks on a 3330, 3330-1, or 3340. A specification of 1 through 255 indicates the number of passes to be made per track for volumes other than a

3330, 3330-1, or 3340 volume. If PASSES is omitted, one pass is made per track.

BYPASS=YES bypasses the program's defective-track checking feature.

If BYPASS is omitted, tracks are checked and those found defective are automatically assigned alternates. This operand is ignored if specified for 3330 devices.

VLD Statement

The VLD statement labels the volume. The format of the VLD statement is:

```
[name]  | VLD  | NEWVOLID=serial
        |      | {,VOLPASS=1}
        |      | {,VOLPASS=0}
        |      | [,OWNERID=xxxxxxxxxx]
        |      | [,ADDLABEL=n]
```

where:

NEWVOLID=serial is a one- to six-character volume serial number.

VOLPASS=1 sets the volume security bit to 1.

VOLPASS=0 sets the volume security bit to 0.

If VOLPASS is omitted, the volume security bit is set to 0.

OWNERID=xxxxxxxxxx is a one- to ten-character field that identifies the owner of the volume.

If OWNERID is omitted, no identification is given.

Note: The ownerid CP370 is reserved for use by DMKFMT and cannot be specified.

ADDLABEL=n is a number between one and seven that indicates the total number of additional labels for which space is to be allocated.

If ADDLABEL is omitted, 0 is assumed.

VTOCD Statement

The VTOCD statement contains information for controlling the location of the volume table of contents. The format of the VTOCD statement is:

```
[name]  |VTOCD  |STRTADR=nnnnn
        |       |,EXTENT=nnnn
```

where:

STRTADR=nnnnn is the one- to five-byte track address, relative to the beginning of the minidisk, at which the volume table of contents is to begin. The VTOC cannot occupy cylinder 0, track 00, or any alternate track.

EXTENT=nnnnn is the length of the volume table of contents in tracks. The number (decimal) of entries per track for each type of device is given below.

Device	VTOC Entries/Track
2301	63
2314/2319	25
2311	16
2305-1	18
2305-2	34
3330-1,2	39
3330-11	39
3340	22

IPLTXT Statement

The IPLTXT statement separates service program control statements from IPL program text statements. It is required only when IPL text is included. The statement consists of the word IPLTXT, followed by blanks.

When IPL text is included, the END statement must follow it and END must start in column 2.

END Statement

The END statement denotes the end of the job. It appears after the last function-defining statement. The format of the END statement is:

```
[name]     END      [user information]
```

LASTCARD Statement

The LASTCARD statement is required only when a IBCDASDI job or a series of stacked IBCDASDI jobs is followed by other statements on the control statement input device. The LASTCARD statement must follow the last END statement applying to an IBCDASDI job. It consists of the operand LASTCARD, followed by blanks.

Example

The example below shows the control statements that might be prepared to initialize minidisk LIBRES for OS data set residence. Note that the label of the real volume, CPVOL1, cannot be used as the VOLID operand.

```
JOB
MSG    TODEV=1403,TOADDR=00E
DADEF  TCDEV=2314,TOADDR=231,VOLID=SCRATCH,CYLNO=50
VLD    NEWVOLID=LIBRES,OWNERID=OPERATIONS
VTOCD  STRTADR=10,EXTENT=5
END
```

The desired size of the minidisk is specified through the CYLNO operand. The value specified includes one cylinder reserved for alternate track assignment; a user assigned n cylinders has n-1 of these initialized for his use, and the nth cylinder used for any alternate track assignment. The minimum size of a minidisk can be computed from the formula below. In it, N represents the minimum number of cylinders, and K represents the number of recording heads of the device. The SIZE-OF-VTOC value should be in tracks, and the result of the division should be rounded to the next highest integer.

```
N = 2 + (SIZE-OF-VTOC / K)
```

Note: For 2305 Models 1 and 2, 3330, and 3340 devices, all n cylinders are assigned for your use. No allowance is made for alternate track assignment.

ASSIGNING AN ALTERNATE TRACK

IBCDASDI: (1) analyzes a track and, if necessary, assigns an alternate or (2) bypasses testing, and assigns an alternate. You must specify the tracks for which you wish alternates with a GETALT statement.

Assigning an Alternate (with Testing): An alternate track (if available) is assigned for a track specified for testing and found defective. If the defective track has had an alternate previously assigned, a new alternate track is assigned. If the defective track is an unassigned alternate track, it is flagged to prevent its future use and another alternate track is selected. The alternate track address is made known to the operator.

If a track is tested and found to be "not defective," no alternate is assigned. The operator is notified by a message.

Assigning an Alternate (without Testing): The program's defective track checking feature can be bypassed, and an alternate track can be assigned for any track, whether it is permanently defective or not. If the specified track is an alternate, a new alternate track is assigned. If the specified track is an unassigned alternate, it is flagged to prevent its future use.

Note: For 3330 and 3340 minidisks you must assign the alternate on the real disk. Any references thereafter to that track on the minidisk are referred to the alternate track.

GETALT Statement

Any number of alternate tracks can be assigned in a single job by including one GETALT statement for each track. The GETALT statement can follow the MSG statement. The format of the GETALT statement is:

```
[name]    GETALT  TODEV=xxxx
                  ,TOADDR=cuu
                  ,TRACK=ccccchhhh
                  ,VOLID=serial
                  [,FLAGTEST=NO]
                  [,PASSES=n]
                  [,BYPASS=YES]
                  [,MODEL=n]
```

where:

TODEV=xxxx is the device type of the direct access device.

TOADDR=cuu is the channel number (c) and unit number (uu) of the direct access device.

TRACK=ccccchhhh is the address of the track for which an alternate is requested; where cccc is the cylinder number and hhhh is the head number. These are hexadecimal numbers.

VOLID=serial is the volume serial number of the minidisk to which an alternate track is to be assigned. If "serial" matches the volume serial number found on this minidisk, the alternate track assignment proceeds. If it does not match, the operator is notified.

FLAGTEST=NO (used when testing before assigning an alternate) specifies that the program not check for a previously flagged track before a surface analysis is attempted on this track (disk storage devices only).

PASSES=n (used when testing before assigning an alternate) specifies that the program's defective track checking feature is to make n number of passes (from 1 to 225) when performing a surface analysis. If PASSES is omitted, one pass is made on this track.

BYPASS=YES is that the program's defective track checking feature is to be bypassed.

If BYPASS is omitted, the program assigns an alternate only if it finds that the specified track is defective.

MODEL=n is a decimal model number (1 or 2). This operand is only for the 2305 and corresponds to the 2305-1 and 2305-2, respectively.

Note: A list of defective tracks (if any) is provided with new IBM disk storage volumes. Refer to this list when using the IBCDASDI program for the first time. After initialization, include the GETALT statement in an IBCDASDI job to assign an alternate track for each track on the list. Subsequent IBCDASDI jobs "remember" those defective tracks, unless the FLAGTEST=NO option is specified for those jobs.

INVOKING THE IBCDASDI PROGRAM

The IBCDASDI program is invoked for minidisks by specifying the operand

```
...,CYLNO=nnn
```

on the DADEF control statement (discussed previously). This control statement is passed to the IBCDASDI utility for processing. This operand specifies the size, in cylinders, of the minidisk to be initiated.

The IBCDASDI program, which is distributed as a CMS file with a filename of IPL and a filetype of IBCDASDI, should be spooled to your own virtual card reader. Control statements for the program can follow the last card or card image for the program, or can be entered via a separate input device.

To execute the IBCDASDI program:

1. Spool a copy of the IBCDASDI object module to your virtual card reader, or mount and attach the tape containing the object program.

2. Load the object program from the virtual reader or tape by issuing the CP IPL command for the appropriate virtual device address. When the program is loaded, an enabled wait state is entered with the address field of the PSW containing the hexadecimal value FFFF.

3. When the program is loaded and waiting for input, signal attention from the virtual console device. The message

   ```
   DEFINE INPUT DEVICE
   ```

   is sent to your virtual console. Enter the following response from the virtual console:

   ```
   INPUT=type,cuu
   ```

   where:

   type — is the device type of the device containing the control statements. Valid device types are 2400, 2540, 3410, 3420, and 3505.

   cuu — is the device address of the device containing the control statements.

Control statements are printed on the message output device. At the end of job, the END OF JOB message is printed on the message output device and the program enters the wait state.

FORMATTING VOLUMES--GENERAL INFORMATION

All direct access volumes used by the VM/370 system (for paging, spooling, system residence, directory, or temporary disk allocation) must be properly labeled, formatted, and allocated. The CP Format/Allocate service program (DMKFMT module) prepares disks for use by CP. A CMS FORMAT program is also available and must be used to format CMS and RSCS disks.

Volumes used only by virtual machines other than CMS or RSCS (minidisks or dedicated volumes) do not need to be formatted.

An object deck version of the CP Format/Allocate service program is a standalone program and can be loaded from a virtual or real card reader in, respectively, a virtual or a real machine. (If run in a virtual machine, the virtual machine must have write access to the volume being formatted.) The program accepts control statements from the operator's system console (commands) or from the IPL device (card reader).

Cylinders used by CP for paging, spooling, and so on, must be preformatted with fixed length unblocked records of 4096 bytes.

Device capacity when formatted for CP use is:

2314/2319	32 records/cylinder	(8 records/5 tracks)
3330	57 records/cylinder	(3 records/track)
2305	24 records/cylinder	(3 records/track)
3340	24 records/cylinder	(2 records/track)

The format operation writes 4096-byte blocks on all cylinders being formatted. The service program does write-checking to verify that parts of the track are not defective. Bad surfaces are flagged as allccated (in use) in the page bit map and are not used. Alternate tracks are not assigned for CP-owned DASD devices.

For example, the 3330 track format for all formatted cylinders except cylinder 0 is shown in Figure 32.

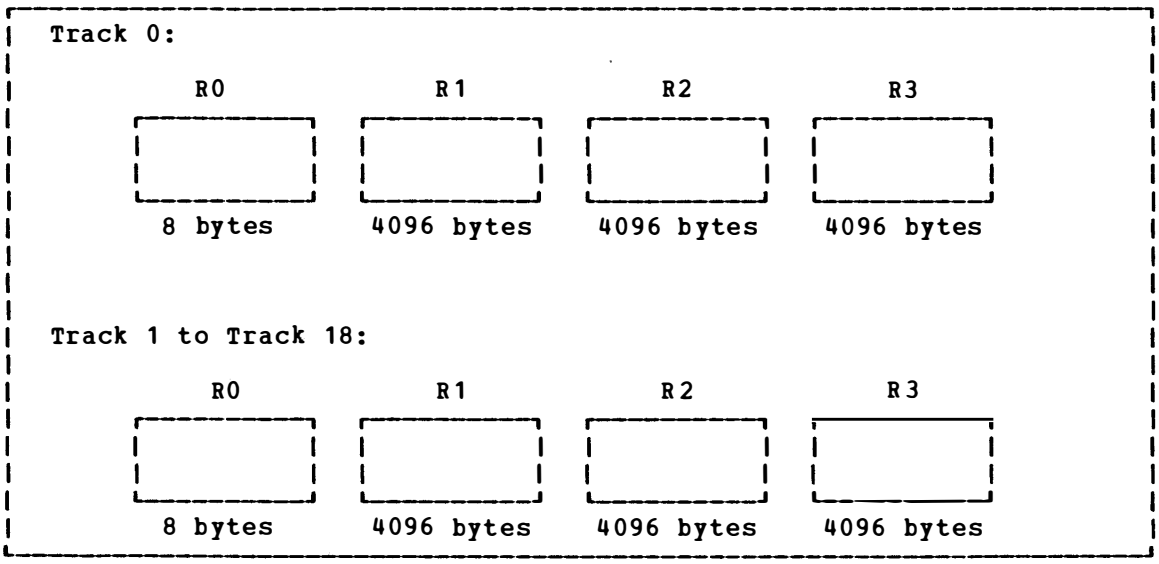


Figure 32. Format of 3330 Cylinders for Use by CP

Cylinder 0 Format

All volumes containing space for CP use (paging, spooling, and so on) must have a properly formatted cylinder 0. The only service program that can do this is the Format/Allocate program (DMKFMT).

Cylinder 0 is formatted like other cylinders except that the space associated with the first three 4096-byte blocks is reserved for system use. This area is then formatted as illustrated in Figure 33.

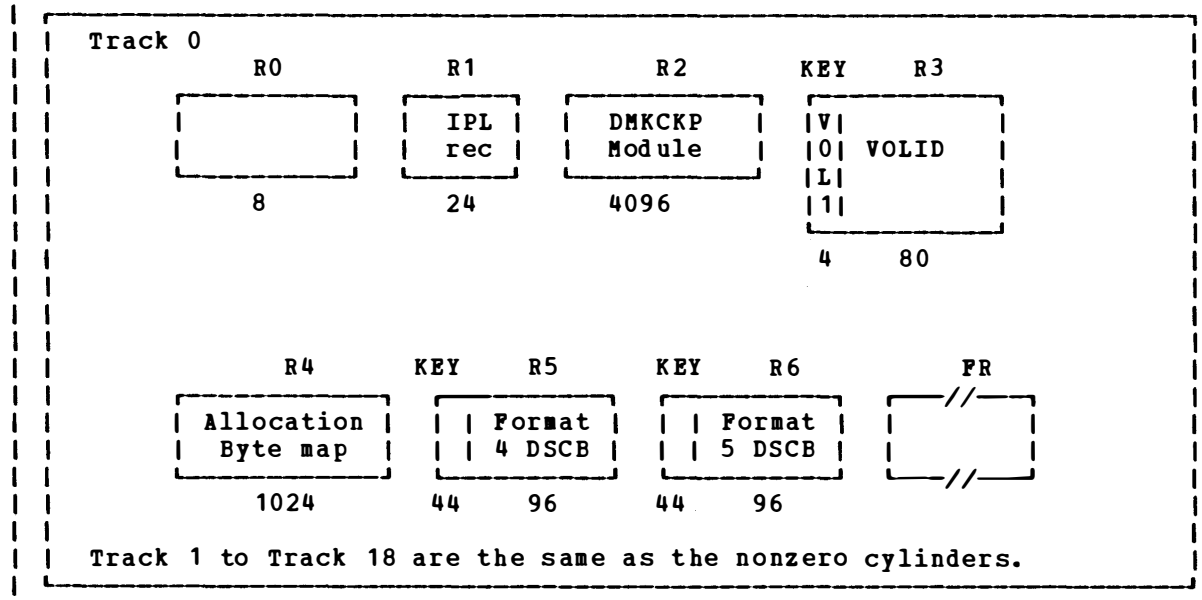


Figure 33. 3330 Cylinder 0 Format

The contents of each record in cylinder zero are as follows:

R0 Nothing.

R1 IPL record -- Puts the system into wait state if storage volume is loaded before CP nucleus is built.

R2 Checkpoint record -- Used by CP to save and retrieve information for a warm start.

R3 Volume label -- Same as OS VOL1 label. On CP system residence volume, area in data record marks the beginning of the system directory. A label is automatically written when cylinder 0 is formatted. The owner field of the label record contains "VM/370" if there is allocation data present in R4.

R4 Allocation Byte Map -- Each byte identifies a cylinder and specifies its usage (paging, spooling, directory, and so on). This map is filled in by the ALLOCATE function of the DMKFMT service program.

R5 Format 4 OS DSCB type label -- For compatibility with OS.

R6 Format 5 OS DSCB type label -- For compatibility with OS. Label indicates to OS that no space is available on this volume.

FR Is one or more filler records.

FORMAT/ALLOCATE SERVICE PROGRAM (DMKFMT)

The Format/Allocate service program formats, allocates, and labels direct-access volumes for paging, spooling, and CP file residence. This service program is executed as part of CP system generation procedures and may also be executed as a standalone program to:

1. Format direct-access volumes for CP use.
2. Allocate specific disk areas to particular functions or to CP use.
3. Write six-character volume serial number labels.

Note: The Format/Allocate program should be used with care, since it destroys existing data (if any). Also, user minidisks must not begin on real cylinder zero of CP-owned volumes, because information critical to CP is stored in that cylinder.

Format/Allocate program control statements may be supplied in card form via a card reader, or may be entered at the system console. All error messages regarding improper specification of control statements are displayed at the console. If the Format/Allocate service program encounters any unusable disk surfaces, it flags the associated record, but it does not select any alternate tracks.

FORMAT/ALLOCATE PROGRAM CARD INPUT

Punch control statements for card input start in column 1, and each field is separated from the adjacent field by a comma. Two commas in a row cause the insertion of a default value. Three commas in a row cause the insertion of two default values.

Note: The only default values permitted are those that define the starting and ending cylinders. The defaults are the first and last cylinders of the volume, respectively.

Comments must be preceded by at least three blanks.

The control card entries for the Format/Allocate program must be in the following order:

- Format function:

```
FORMAT,devadr,devtype,volser,startcyladr,endcyladr
```

- Allocate function:

```
ALLOCATE,devadr,devtype,volser
TEMP,startcyladr,endcyladr
PERM,startcyladr,endcyladr
TDSK,startcyladr,endcyladr
DRCT,startcyladr,endcyladr
END
```

- Label functions:

```
FORMAT,devadr,devtype,volser,LABEL
```

FORMAT, ALLOCATE, and LABEL are Format/Allocate program control words and may be abbreviated to one letter.

FORMAT Control Statement

The format of the FORMAT control statement is:

```
FORMAT,devadr,devtype,volser,startcyladr,endcyladr
```

where:

devadr — is a three-digit hexadecimal number that identifies the address of the device that the Format/Allocate program is to act upon.

devtype — is a four- to seven-character field that defines an approved device for the Format/Allocate program. Approved device types are 2314, 2319, 3330, 3330-11, 3340-35, 3340-70, 2305-1, and 2305-2. Specify 3333 as 3330, and 3340-70F as 3340-70.

volser — is a one- to six-character field that represents the volume serial number of the volume you are formatting.

startcyladr — is the starting cylinder address on the DASD on which the format function is to be performed. The start cylinder address is entered as decimal digits.

endcyladr — is the last cylinder address on the DASD cn which the format function is to be performed. The end cylinder address is entered as decimal digits.

Note: FORMAT is a control word and maybe abbreviated to F.

ALLOCATE Control Statements

The formats of the ALLOCATE control statements are:

```
ALLOCATE,devadr,devtype,volser
TEMP,startcyladr,endcyladr
PERM,startcyladr,endcyladr
TDSK,startcyladr,endcyladr
DRCT,startcyladr,endcyladr
END
```

where:

devadr — is a three-digit hexadecimal number that identifies the address of the device that the program Format/Allocate is to act upon.

devtype — is a four- or seven-character field that defines an approved device for the Format/Allocate program. Approved device types are 2314, 2319, 3330, 3330-11, 3340, 2305-1, and 2305-2.

| volser | is a one- to six-character field that represents the volume serial number of the volume you are formatting.

| startcyladr | is the starting cylinder address on the DASD on which the format function is to be performed. The start cylinder address is entered as decimal digits.

| endcyladr | is the last cylinder address on the DASD on which the format function is to be performed. The end cylinder address is entered as decimal digits.

| TEMP | indicates that the following operands identify temporary storage space reserved for spooling or paging activity.

| PERM | defines an area that can contain the logout area, the CP nucleus, and space that is not used by the system but is available for use by virtual machine users (for example, for user minidisks).

| TDSK | defines the pooled space available for virtual machine users after they have logged on the VM/370 system.

DRCT | indicates that the following cylinders are reserved for directory files.

ALLOCATE is a control word and may be abbreviated to its first letter, A.

TEMP, PERM, TDSK, and DRCT are all functions of ALLOCATE. These cards can follow the ALLOCATE control statement in any sequence. Each card in turn overlays the cylinder table, and any space not reallocated remains the same. If an ALLOCATE function overlays the previous cylinder allotment, then the previous cylinder space allotment is truncated to the beginning of the next cylinder allotment. For example:

Disk Storage Allocation	First Cylinder	Last Cylinder
1st Entry TEMP	000	202
2nd Entry PERM	010	050
3rd Entry TDISK	040	050
4th Entry DRCT	000	004
5th Entry END		

The result of this disk volume allocation is:

Disk Storage Allocation	First Cylinder	Last Cylinder
DRCT	000	004
TEMP	005	009
PERM	010	039
TDSK	040	050
TEMP	051	202

Once an ALLOCATE control statement is encountered, all cards following it until an END card is encountered are assumed to be part of a single allocation. The Format/Allocate service functions cannot be performed on another disk volume until the END card is encountered.

Note: Reallocation of a directory cylinder containing an active VM/370 directory deallocates the directory to allow a new directory to be | written on the same cylinder.

LABEL Control Statement

The format of the LABEL control statement is:

```
FORMAT,devadr,devtype,volser,LABEL
```

where:

devadr — is a three-digit hexadecimal number that identifies the address of the device that the Format/Allocate program is to act upon.

devtype — is a four- or seven-character field that defines an approved device for the Format/Allocate program. Approved device types are 2314, 2319, 3330, 3330-11, 3340, 3340-35, 3340-70, 2305-1, and 2305-2.

volser — is a one- to six-character field that represents the volume serial number.

LABEL — is a keyword designating the label function of the Format/Allocate program.

Note: FORMAT and LABEL are control words and may be abbreviated to F and L, respectively.

Examples:

FORMAT:

```
FORMAT,232,3330,MYDISK,000,006
FORMAT,232,3330,MYDISK,,,
FORMAT,232,3330,MYDISK,,00
FORMAT,232,3330,MYDISK,001,,
```

ALLOCATE:

```
ALLOCATE,232,3330,MYDISK
TEMP,000,050
PERM,055,060
TDSK,100,108
DRCT,110,120
END
```

LABEL:

```
F,232,3330,MYDISK,label
```

FORMAT/ALLOCATE CONSOLE INPUT

The Format/Allocate program can be controlled by control statements entered into the real or virtual console instead of by a deck of cards containing control statements. If the program finds no control statements at the card reader, it issues a prompting message to the console. The proper response causes the prompting message for the next operand to appear until the Format, Allocate, or Label function is completely defined; then the Format/Allocate program is executed.

After execution, the prompting begins again until all DASD allocation requirements are fulfilled.

The sequence for console typewriter processing of the Format/Allocate program (after making the operator's console ready), is as follows:

1. Load the card reader with a loader, followed by the Format/Allocate deck.
2. IPL the card reader.
3. Respond to the first message displayed at the system console.
4. Respond to other messages.

Following are examples of Format/Allocate program execution under CP control. Figure 34 is an example of the label operation, Figure 35 is an example of the allocate operation, and Figure 36 of the allocate overlap operation. All responses are entered after the colon; after a function is complete, the program returns to 'ENTER "FORMAT" OR "ALLOCATE":'

```
VM/370 FORMAT/ALLOCATE PROGRAM VERSION 2.0

ENTER "FORMAT" OR "ALLOCATE":f
FORMAT FUNCTION SELECTED
ENTER DEVICE ADDRESS (CCU):131
ENTER DEVICE TYPE: 2314
ENTER START CYLINDER (XXX) OR "LABEL":l
ENTER DEVICE LABEL:cpdsk2
```

Figure 34. Using the Format Program Label Function

```
ENTER "FORMAT" OR "ALLOCATE":a
ALLOCATE FUNCTION SELECTED
ENTER DEVICE ADDRESS (CCU):131
ENTER DEVICE TYPE: 2314
ENTER DEVICE LABEL:cpdsk2
ENTER ALLOCATION DATA FOR VOLUME CPDSK2
TYPE CYL CYL
.... ... ...
drct 000 001
perm 004 008
tdsk 100 150
end
ALLOCATION RESULTS
DRCT 000 001
TEMP 002 003
PERM 004 008
TEMP 009 099
TDSK 100 150
TEMP 151 202
DEVICE 131 VOLUME CPDSK2 ALLOCATION ENDED
```

Figure 35. Using the Format Program Allocate Function

```
ENTER "FORMAT" OR "ALLOCATE":a
ALLOCATE FUNCTION SELECTED
ENTER DEVICE ADDRESS (CCU):131
ENTER DEVICE TYPE: 2314
ENTER DEVICE LABEL: cpdsk2
ENTER ALLOCATION DATA FOR VOLUME CPDSK2
TYPE CYL CYL
.... ... ...
perm 004 004
temp 000 010
tdsk 000 010
perm 010 202
end
ALLOCATION RESULTS
TDSK 000 009
PERM 010 202
DEVICE 131 VOLUME CPDSK2 ALLOCATION ENDED
```

Figure 36. Using the Format Program Allocate Overlap Function

Note that before the ALLOCATE function was invoked, cylinder 0 was formatted and labeled CPDSK2. The area associated with the first three 4096-byte blocks on cylinder 0 are not used for spooling but contain system information (page allocation map, label, and so on).

These CP-formatted volumes can be made usable by CP in one of two ways:

1. They may be attached to the system by the VM/370 operator.

2. Their volume serial numbers may appear in the SYSOWN macro in the DMKSYS module. The CP system residence volume's serial number must appear in the SYSOWN macro.

Appendix G: Compatibility of VM/370 with CP-67/CMS

VM/370 and its components, the control program (CP) and the Conversational Monitor System (CMS), are based on the CP-67/CMS system, and are designed especially for the IBM . The Dynamic Translation Facility on the System/370 System/370 provides the same facilities as did Dynamic Address Translation (DAT) on the System/360 Model 67, but differs in hardware design details and software implementation. Consequently, the CP-67/CMS system does not run on a System/370, and the VM/370 system does not run on the System/360 Model 67. The internal control block structure differs between the two systems, and user modifications to the CP-67/CMS system may no longer be necessary, desirable, or usable in the new system without some redesign effort.

Certain commands familiar to CP-67 users are not supported in VM/370. The functions available under CP-67 do, however, have equivalents in VM/370 where appropriate. The following is a list of all CP-67 console functions, with operand variations, showing the incompatibilities with VM/370. Functions listed under CP-67 Version 3.1 and not under VM/370 indicate that the syntax and function are identical. There are, of course, additional functions available with the VM/370 commands. The CP and CMS commands are described in the VM/370: Command Language Guide for General Users and in the VM/370: Operator's Guide.

Status:
- A - CP-67 — Syntax accepted in VM/370
- R - CP-67 — Syntax not accepted - Functionally supported by another command format.
- N - CP-67 — Syntax not accepted - function not supported in VM/370.

In Figure 37, the letter in the Status column has the following meanings:

- A indicates CP-67 syntax accepted in VM/370
- R indicates CP-67 syntax not accepted but the function is supported by a different VM/370 command
- N indicates CP-67 syntax not accepted and the function is not supported in VM/370.

Whenever an R appears in the "Status" column, the VM/370 command that provides the same function as a former CP-67 command is shown in the "VM/370 Equivalent" column.

Status	CP-67 Command	VM/370 Equivalent
R	ACNT	ACNT ALL
A	ATTACH cuu TO {SYSTEM / userid} AS {volid / vaddr}	
A	BEGIN	
A	BEGIN hexloc	
A	DCP hexloc1[-hexloc2]	
A	DCP Lhexloc1[-hexloc2]	
A	DCP Thexloc1[-hexloc2]	
A	DETACH vaddr	
R	DETACH raddr	DETACH raddr FROM {SYSTEM / userid}
A	DIAL system	
N	DIRECT LOCK	
N	DIRECT UNLOCK	
A	DISABLE line	
A	DISABLE ALL	
A	DISCONN	
R	DISCONN xxx	DISCONN HOLD
A	DISPLAY hexloc1 [-hexloc2]	
R	DISPLAY L[-hexloc2]	DISPLAY L0 [-hexloc2 / -END]
A	DISPLAY Lhexloc1[-hexloc2]	
R	DISPLAY T[-hexloc2]	DISPLAY T0 [-hexloc2 / -END]
A	DISPLAY Thexloc1[-hexloc2]	

Figure 37. VM/370 Compatibility with CP-67 (Part 1 of 4)

Status	CP-67 Command	VM/370 Equivalent
R	DISPLAY K[-hexloc2]	DISPLAY K0[-hexloc2 | -END]
A	DISPLAY Khexloc1[-hexloc2]	
A	DISPLAY G[-reg2]	
A	DISPLAY Greg[-reg2]	
A	DISPLAY Y[-reg2]	
A	DISPLAY Yreg[-reg2]	
A	DISPLAY X[-reg2]	
A	DISPLAY Xreg[-reg2]	
A	DISPLAY PSW	
A	DMCP hexloc1[-hexloc2]	
R	DMCP L[-hexloc2]	DMCP L0[-hexloc2 | -END]
A	DMCP Lhexloc1[-hexloc2]	
R	DMCP T[-hexloc2]	DMCP T0[-hexloc2 | -END]
A	DMCP Thexloc1[-hexloc2]	
A	DRAIN	
N	DUMP	
A	ENABLE line	
A	ENABLE ALL	
A	EXTERNAL	
A	IPL CMS	
A	IPL devadd	
R	IPLSAVE CCW	IPL vaddr NOCLEAR
R	KILL userid	FORCE userid
N	KILL	
A	LINK userid xxx yyy [W | R] [PASS=pwd]	
R	LINK userid xxx yyy [W | R] (NOPASS)	LINK userid xxx yyy [W | R]
A	LOCK userid fpage lpage	
A	LOGIN userid	
R	LOGIN userid xxx	LOGON userid MASK
A	LOGOUT	
R	LOGOUT xxx	LOGOFF HOLD

Figure 37. Compatibility of VM/370 with CP-67 (Part 2 of 4)

Status	CP-67 Command	VM/370 Equivalent
A	MSG userid line	
R	MSG CP line	MSG OPERATOR line
A	MSG ALL line	
R	PSWRESTART	SYSTEM RESTART
A	PURGE READER	
A	PURGE PRINTER	
A	PURGE PUNCH	
R	QUERY DEVICE ALL	QUERY ALL
R	QUERY DEVICE xxx	QUERY xxx
A	QUERY DUMP	
A	QUERY FILES	
A	QUERY LOGMSG	
N	QUERY MAX	
A	QUERY NAMES	
n	QUERY PORTS	
R	QUERY PORTS ALL	QUERY LINES
N	QUERY PORTS FREE	
R	QUERY PORTS xxx	QUERY xxx
A	QUERY PRIORITY userid	
R	QUERY Q2	QUERY PAGING
A	QUERY TIME	
A	QUERY userid	
A	QUERY USERS	
A	QUERY VIRTUAL xxx	
A	QUERY VIRTUAL CORE	QUERY VIRTUAL STORAGE
A	QUERY VIRTUAL ALL	
A	READY xxx	
A	REPEAT xxx y	
R	RESET	SYSTEM RESET
R	SET ADSTOP xxxxxx	ADSTOP xxxxxx
R	SET ADSTOP OFF	ADSTOP OFF
R	SET APLBALL {ON / OFF}	TERMINAL APL {ON / OFF}
R	SET ATTN {ON / OFF}	TERMINAL ATTN {CN / OFF}
R	SET CARDSAVE {ON / OFF}	SPOOL READER {HOLD / NOHOLD}
A	SET DUMP xxx	
A	SET {LINEDIT / RUN} {ON / OFF}	
A	SET LOGMSG	
A	SET LOGMSG NULL	
A	SET LOGMSG n	

Figure 37. Compatibility of VM/370 with CP-67 (Part 3 of 4)

Status	CP-67 Command	VM/370 Equivalent
N	SET MAX	
A	SET MSG { ON / OFF }	
R	SET Q2 nn	SET PAGING nn
R	SET TRACE devtype	TRACE type dev
R	SET TRACE OFF	TRACE type OFF
R	SET TRACE END	TRACE END
A	SET WNG { ON / OFF }	
A	SHUTDOWN	
A	SLEEP	
A	SPACE xxx	
R	SPOOL xxx ON yyy	SPOOL yyy CLASS x
A	SPOOL xxx CONT	
R	SPOOL xxx OFF	SPOOL xxx NOCONT
A	START xxx	
A	STCP hexloc	
A	STCP Shexloc	
A	STORE Lhexloc hexinfo...	
A	STORE Shexloc hexstring	
A	STORE Greg hexinfo...	
A	STORE Yreg hexinfo...	
A	STORE Xreg hexinfo...	
A	STORE PSW [hexinfo1] hexinfo2	
R	TERM xxx	FLUSH xxx
A	UNLOCK userid fpage lpage	
A	WNG userid text	
A	WNG ALL text	
R	XFER xxx { TO userid / OFF }	SPOOL xxx { TO userid / OFF }

Figure 37. Compatibility of VM/370 with CP-67 (Part 4 of 4)

Incompatibility Statement to CP-67/CMS Users

Although the CMS in VM/370 is built upon CMS Version 3.1 in CP-67/CMS, there are five types of modifications that were made to 3.1 that affect the relationship between versions:

1. Unchanged: Some commands and system functions remain unchanged; therefore, complete compatibility exists.

2. Additional Functional Capability: Functional and syntactical enhancements are effected; but, in some cases, old keywords and functions are supported.

3. Command Name Alterations: Commands have name changes, but a SYNONYM file may be included during nucleus system generation.

4. Keyword Changes: Some keywords within a command are modified, deleted or added.

5. Major Modifications: Improvements to commands and system functions caused complete incompatibilities in the following areas:

   - Because the CMS nucleus is significantly larger, all MODULES must be recreated from their object (TEXT) files using the GENMOD command.

   - Because you may now have up to 10 disks, the logical directory identifications (filemode letters) are changed to reflect a more natural, easy-to-remember, search order: P, T, A, B, S, C becomes A, B, C, D, E, F, G, S, Y, Z -- with the system disk being the S-disk, and the primary disk becoming the A-disk.

   - The following global changes of filetypes must be made:

     ```
     SYSIN  to ASSEMBLE
     ASP360 to MACRO
     ```

   - For language processors, the DECK and NODECK options have a new meaning, they route the object (TEXT) file to the spooled card punch; the LOAD and NOLOAD options now invoke the function formerly performed by DECK and NODECK, and the writing of the TEXT file onto a CMS disk.

   - No 2311 disk support is provided for CP or CMS files.

   - In Version 1.0 the tape designations are as follows:

     ```
     TAP1 is for 181
     TAP2 is for 182
     ```

     The default for tape commands is TAP1.

- CMS does not function on a real CPU without the control program.
- TXTLIB files must be recreated.
- Because many fields are changed in the CMS nucleus or rearranged, many of your programs that refer to these fields have to be reassembled with the new CMS macro libraries.
- Modules that refer to fields containing sizes, limits, and quantities within the CMS nucleus might have to be reassembled and then regenerated.
- SYSLIB MACLIB is renamed to CMSLIB MACLIB.
- All EXEC files should be checked for command name and operand changes, filemode usage, and so on, and changed to conform to VM/370 CMS. Major changes are:

```
&TYPEOUT to &CONTROL
&PRINT   to &TYPE
&INDEX0  to &RETCODE
```

- The options specified for a LOAD command do not remain in effect for subsequent INCLUDE commands; options are reset to default settings unless the SAME option is specified.
- Filenames and filetypes must be composed entirely of alphameric characters.
- If the CMS Version 1.0 system does not recognize a command name, the command line is automatically passed to CP. If the CMS SET and QUERY commands do not recognize an operand or option, the command line is passed to CP. This is not true for EXEC files; the CP command must be explicitly stated. The feature may be negated by entering the command:

```
SET IMPCP OFF
```

VM/370 CMS SUPPORT OF CP-67/CMS COMMANDS

ALTER

Command name changed to RENAME.
Components of the new file identifier may not be specified as asterisk (*). An equal sign (=) performs the same function.
NOUP option keyword changed to NOUPDIRT; NOUP is the abbreviation.
Default options added: NOTYPE, UPDIRT.

ASSEMBLE

Only one file may be assembled per ASSEMBLE command.

Options Changed:
NODECK is the default
DIAG|NODIAG changed to TERM|NOTERM
LTAPn not supported
LDISK option name changed to DISK

Options Added:
LOAD|NOLOAD, ALGN|NOALGN
OS|DOS, TEST|NOTEST, LINECT nn|55,
NUM|NONUM, STMT|NOSTMT

BLIP	Functionally supported by SET BLIP.
BRUIN	Not implemented.
CEDIT	Functionally supported by EDIT.
CHARDEF	Functionally supported by CP TERMINAL CHARDEF command.
CLOSIO	Functionally supported by the CP commands, SPOOL and CLOSE.
CLROVER	Functionally supported by SVCTRACE OFF command.
CNVT26	Functionally supported by COPYFILE command with EBCDIC option.
COMBINE	Functionally supported by COPYFILE.
COMPARE	Filemode required. NCSEQ option functionally supported by COL opticn.
CPFUNCTN	Command name changed to CP. NOMSG option not supported.
CVTFV	Functionally supported by COPYFILE.
DEBUG	No change.

DEBUG Subcommands

BREAK	No change.
CAW	No change.
CSW	No change.
DEF	Subcommand name changed to DEFINE; DEF minimum truncation; no change in format.
DUMP	No change.
GO	No change.
GPR	No change.
IPL	Not supported from DEBUG.
KX	Supported by CMS command HX.
ORIGIN	No change.
PSW	No change.
RESTART	Not supported.
RETURN	No change.
SET	No change.
STORE	No change.
TIN	Not supported.
X	If symbol specified, length of field defaults to 4.

DISK — No change.

DUMPD — Functionally supported by DDR command.

DUMPF — Functionally supported by TYPE command with HEX option.

DUMPREST — Functionally supported by DDR command.

ECHO — Functionally supported by CP command ECHO.

EDIT — Filename must be specified.
Filemode may be specified.
LRECL may be specified for a new file.

EDIT Subcommands

BACKSPACE — Functionally supported by CMS command, SET INPUT.

BLANK — Functionally supported by OVERLAY.

BOTTOM — No change.

BRIEF — Function accomplished by VERIFY OFF request.

CHANGE — No change.

DELETE — /string/ is not valid as an operand.
(DELETE * may be used to delete to end of file.)

FILE — Filetype and filemode may be specified.

FIND — No change.

INPUT — No change.

INSERT — Functionally supported by INPUT request.

LOCATE — No change.

NEXT — No change.

OVERLAY — No change.

PRINT — Request name changed to TYPE.
(TYPE * may be specified to indicate that typing is to continue until EOF.)
Instead of L in second field, * is specified.

QUIT — No change.

REPEAT — Valid only for subsequent OVERLAY subcommand.
(REPEAT * may be specified to indicate that the OVERLAY subcommand is to be repeated for the remainder of the file.)

RETYPE	Request name changed to REPLACE.
SAVE	Filetype and filemode may be specified.
SERIAL	First operand changed to {OFF|'seq'(same as ID)|ON|ALL}.
TABDEF	Functionally supported by CMS command SET INPUT.
TABSET	No change.
TCP	No change.
UP	No change.
VERIFY	Operand format changed; function added.
X	No change.
Y	No change.
ZONE	No change.

ERASE — Default option added; old format accepted.
* * * Not supported.

EXEC — No change.

EXEC Control Words

&ERROR	Action does not default to &CONTINUE.
&IF	No change.
&EXIT	No change.
&QUIT	Functionally supported by &EXIT 0.
&SKIP	No change.
&GOTO	EXIT not supported as operand. Line-number now a valid operand.
&LOOP	No change.
&CONTINUE	No change.
&TYPEOUT	Functionally supported by &CONTROL control word; ON, NOEXEC, RESUME, KILL not valid. CMS operand added.
&TIME	Operands changed to [ON | OFF] [RESET | TYPE].
&SPACE	No change.
&PRINT	Functionally supported by &TYPE.
&UPRINT	Functionally supported by &BEGTYPE.

&PUNCH	No change.
&UPUNCH	Functionally supported by &BEGPUNCH.
&COMMENT	Functionally supported by * card.
&ARGS	No change.
&READ	VARS operand added.
&STACK	No change.
&BEGSTACK	ALL operand added.
&ENDSTACK	Functionally supported by &END.
&SET	Not implemented.

FILEDEF Device names changed:

CON	– TERMINAL
DSK	– DISK
DSK–nn	– not supported
DUMMY	– no change
PRT	– PRINTER
PUN	– PUNCH
RDR	– READER
TAPEn	– unchanged
BAT	– not supported

FINIS Not supported from terminal.

FORMAT All functions supported; all operands changed.

FORTRAN This command is now supported by the commands FORTG1, FORTHX, GOFORT, and CONVERT which invoke IBM Program Products.

GENDIRT Target mode parameter is added.

GENMOD Incompatible; positional parameters and opticns changed; equivalent function performed.

GLOBAL PRINT function removed; others compatible.

IPL Not explicitly supported in CMS; however, the command may be issued and is passed to the control program for processing.
Either a device address or system name must be specified.
Cyl–no may be specified.

Options Added: CLEAR|NOCLEAR.

KE Functionally supported by the CP command TERMINAL LINESIZE nn.

KO Command name changed to HO.

KT Command name changed to HT.

KX Command name changed to HX.

LINEND Functionally supported by the CP command TERMINAL LINEND.

LISTF — Command name changed to LISTFILE; LISTF accepted.

Option Changes:
- SORT- Option not implemented.
- ITEM- Option not implemented; ALLOC option produces both logical records and blocks.
- NAME- Option name changed to FNAME.
- TYPE- Option name changed to FTYPE.
- MODE- Option name changed to FMODE.
- REC- Supported by ALLOC option.
- DATE- Produces mm/dd/yy hh:mm.
- YEAR- Not implemented; functionally supported by DATE option.
- TIME- Not implemented; functionally supported by DATE option.

LABEL and FORMAT options added.
APPEND option added.
HEADER|NOHEADER option added.

LOAD

Option Changes:
- SLCxxxxxx- option changed to ORIGIN xxxxxx.
- SLC12000- default changed to first available location.
- SINV- option name changed to NOINV.
- PINV- option name changed to INV.
- SREP- option name changed to NOREP.
- PREP- option name changed to REP.
- SLIBE- option name changed to NOLIBE.
- SAUTO- option name changed to NOAUTO.
- XEQ- option name changed to START.
- NOXEQ- option not supported.

Options Added: RESET.

TXTLIB files may no longer be specified in a LOAD command, but must have been previously specified by a GLOBAL command.

LOADMOD — Filetype must be specified if filemode is given.

LOGIN — Functionally supported by LOGON and ACCESS commands. Comma between mode and extdisk replaced with slash (/); extdisk optional.

Options:
- NOPROF-option supported.
- NOTYPE-option not supported.
- NO-UFD-option name changed to ERASE.

LOGOUT — Functionally supported by CP command LOGOFF.

MAPPRT — Functionally supported by CMS commands TYPE or PRINT.

MODMAP — No change.

OFFLINE READ — Functionally supported by READCARD command.

OFFLINE PRINT — Functionally supported by PRINT command.

OFFLINE PRINTCC Functionally supported by PRINT command.

OFFLINE PRINTUPC Functionally supported by PRINT command.

OFFLINE PUNCH   Functionally supported by PUNCH command.

OFFLINE PUNCHCC Functionally supported by PUNCH command.

OFFLINE PUNCHDT Functionally supported by PUNCH command.

OSTAPE Command name Changed to TAPPDS.

PLI This command is now supported by the command PLIOPT which invokes an IBM Program Product.

PRINTF Command name changed to TYPE.
Filemode may be specified.
n3 functionally supported by COL option.

Options added: HEX, MEMBER.

RELEASE DET option not implemented although functionally supported by the CP command DETACH.

REUSE Command name changed to INCLUDE.
Option differences are the same as for LOAD.
TXTLIB files may no longer be specified in the command; but must have been previously specified by a GLOBAL command.

RT No change.

SCRIPT Files may be processed by SCRIPT/370, an IBM user-installed program.

SETERR Functionally supported by SVCTRACE ON.

SETOVER Functionally supported by SVCTRACE ON.

SNOBOL Not implemented.

SORT Filemode must be specified for both input and output files.

SPLIT Functionally supported by COPYFILE command.

START (NO) operand not valid.

STAT Functionally supported by QUERY command.

STATE No change.

SYN — Command name changed to SYNONYM; SYN minimum truncation. Filetype must be SYNONYM.

Options:
SYNONYM command with P and PUSER options is functionally supported by QUERY SYNONYM.

TAPE — TAPn may also be specified by a virtual address. "n" options after SCAN, SKIP, SLOAD, replaced by 'EOF n'.

Options Added: 7TRACK|9TRACK, DEN, TRTCH.

Functions Added: MODESET, BSF, BSR, ERG, FSF, FSR, RUN, REW.

TAPE DUMP — Options added: WTM|NOWTM, NOPRINT|PRINT|TERM|DISK.

TAPE LOAD — Functionally supported by TAPE LOAD EOFn. File identifiers may be specified.

Options Added: NOPRINT|PRINT|TERM|DISK, EOFn|EOT|EOF1.

TAPE SCAN — Filename and filetype may be specified. Functionally supported by TAPE SCAN (EOF n).

Options added: NOPRINT|PRINT|TERM|DISK, EOFn|EOT|EOF1.

TAPE SKIP — Comments same as for TAPE SCAN.

TAPE SLOAD — Functionally supported by TAPE LOAD fn ft. Filemode may be specified.

TAPEIO — Functionally supported by TAPE.

TAPPDS — Default filename is TAPPDS for NOPDS option. Default filetype is CMSUT1.

Option Changes:
- NPDS — option name changed to NOPDS.
- NCOLI — option name changed to NOCOLI.
- TAPx — default is TAP1.
- NEND — option name changed to NOEND.
- NMAXTEN — option name changed to NOMAXTEN.

TAPRINT — Functionally supported by MOVEFILE command.

TPCOPY — Functionally supported by MOVEFILE command.

TXTLIB — PRINT and LIST functions supported by MAP function.

UPDATE — Option Changes:
P — option name changed to REP.
Default options added NOREP, NOSEQ8, NOINC.

USE — Functionally supported by INCLUDE command with the SAME option. See discussion of REUSE compatibility.

VSET — Command name changed to SET.

Functions:

BLIP	ON may be specified to return to default
CHARDEF	Functionally supported by CP command TERMINAL|{CHARDEL|LINEDEL|ESCAPE}
IMPEX	No change.
LDRTBLS	No change.
LINEND	Functionally supported by CP command TERMINAL LINEND.
RDYMSG	ON|OFF changed to LMSG|SMSG.
REDTYPE	No change.
RELPAGE	No change.

Functions added: INPUT, OUTPUT, ABBREV, IMPCP.

WRTAPE — Functionally supported by TAPE command or MOVEFILE command.

$ — Command name changed to RUN.
Filetype and filemode may be specified.

Files may also have filetypes in addition to EXEC, MODULE, and TEXT of those used by the language prccessors for input.

CP-67/CMS MACROS AND FUNCTIONS AND CORRESPONDING CMS MACROS

The list below shows VM/370 CMS macros that correspond to CP-67/CMS macros and functions. The CP-67/CMS functicns have no structural equivalent in VM/370 CMS, but in most cases the function is available via a VM/370 CMS macro. If you need information on how to build a parameter list which can be scanned properly by VM/370 CMS, refer to the VM/370: System Programmer's Guide or VM/370 Command Language Guide for General Users.

CP-67/CMS Macros	VM/370 Equivalent	CP-67 Functions	VM/370 Macros Equivalent
CKEOF	Not Available	DEBDUMP	Not Available
CMSREG	REGEQU	ERASE	FSERASE[1]
CMSYSREF	Not Available	FILEDEF	[1]
ERASE	FSERASE	FINIS	FSCLOSE
FCB	No Change	HNDINT	HNDINT[1]
FINIS	FSCLOSE	HNDSVC	HNDSVC[1]
FSTB	No Change	POINT	FSCB, FSREAD, FSWRITE, FSOPEN
RDBUF	FSREAD		
SETUP	FSOPEN	PRINTR	PRINTL
STATE	FSSTATE	RDBUF	FSREAD
TYPE	WRTERM	STATE	FSSTATE
TYPIN	RDTERM	STRINIT	No Change
WRBUF	FSWRITE	TAPEIO	RDTAPE, WRTAPE, TAPECTL[1]
ATTN	No Change	TRAP	HNDEXT
CARDIO	PUNCH,RDCARD	TYPE	WRTERM[1]
CLOSIO	[1]	TYPLIN	WRTERM
CONWAIT	WAITT	WAIT	WAITD
CPFUNCTN	[1]	WAITRD	RDTERM
		WRBUF	FSWRITE

[1]See the VM/370 System Programmer's Guide for information on how to call a command from a program.

Appendix H: Creating a 3340 System Residence Volume from a Current 2314 or 3330 System Residence Volume and the PLC Tape

VM/370 strongly recommends that you generate the 3340 system residence volume from the 3340 starter system. The process of generating a 3340 system residence volume from the PLC tape and a Release 2 2314 or 3330 system residence volume is complex and restricting; it requires that you generate the CP and CMS nucleus twice. However, you can generate a 3340 system residence volume without using the 3340 starter system. The procedure that follows is one way to do it. You should evaluate the usefulness of the following procedure for your installation before using it.

Before you perform the following procedure, be sure that you have:

- The PLC tape.
- A Release 2 2314/3330 system residence volume.
- A Release 2 2314/3330 system residence volume (CPV2L0) which contains the minidisk areas for updating CP and CMS. These minidisks, 190, 191, 194, and 199, belong to the MAINT virtual machine.
- Two 3340 disks.

Also, you need all of the facilities of VM/370 normally required to update or generate VM/370.

The following procedure assumes your current system residence volume is labeled VMREL2 and the starter system volume (CPV2L0) is set up according to the entries supplied with the Release 2 starter system directories.

STEP 1. IPL CMS

Log on the software system support virtual machine and IPL CMS.

```
logon maint
ipl 190
```

STEP 2. LOAD NEW CP MACRO LIBRARY

Mount the PLC tape and attach it to MAINT as 181. Assuming the PLC tape is at real address 280, you enter:

```
attach 280 to maint as 181
```

Forward space the PLC tape to the DMKMAC MACLIB file and load that file onto the CP staging area minidisk.

```
access 194 a
tape rew
tape fsf n
tape load
```

STEP 3. UPDATE THE REAL I/O CONFIGURATION (DMKRIO) FILE

Edit the DMKRIO ASSEMBLE file and add RDEVICE and RCTLUNIT macros to define the 3340 disks you wish to add to your configuration. Because 3340 disks are now supported by VM/370, omit the CLASS=DASD operand.

Use the VMFASM EXEC procedure to assemble the updated DMKRIO ASSEMBLE file:

```
vmfasm dmkrio dmkr20
```

STEP 4. UPDATE CP AND CMS WITH THE PLC CHANGES

You must rewind the PLC tape and invoke the VMFBLD EXEC procedure to apply the PLC updates to CP and CMS.

First, rewind the tape and load VMFBLD.

```
access 191 a
tape rew
tape load
```

Next, invoke VMFBLD to build a new CP nucleus. Reply to the prompting messages as shown:

```
access 191 c
vmfbld cp
ENTER CP STAGING AREA DISK ADDRESS:
194
ARE YOU A NEW USER? --  RESPOND (YES|NO)
no
GENERATE AN IPL'ABLE SYSTEM TAPE? -- RESPOND (YES|NO)
no
WRITE THE NUCLEUS TO THE SYSTEM DISK? -- RESPOND (YES|NO)
yes
```

VMFBLD builds and loads the CP nucleus and instructs you to shutdown and IPL the new system. If you are building a CP nucleus with a virtual=real area, see the "Using the VMFBLD EXEC Procedure to Build a New Nucleus" section of Part 6 for special considerations and operating procedures.

After you IPL the new system, log on as MAINT, and IPL CMS, rewind the PLC tape and invoke VMFBLD to update CMS.

```
logon maint
attach 280 to maint as 181
ipl 190
access 191 a
tape rew
tape load
access 191 c
vmfbld cms
ENTER CMS SYSTEM DISK ADDRESS:
190
```

Respond to the prompting messages as you normally do when you apply PLC updates.

STEP 5. FORMAT, LABEL, AND ALLOCATE THE 3340 VOLUMES

You must format, allocate, and label two 3340 volumes: one volume to be the system residence volume (VMREL2) and one volume to contain the required minidisks (CPV2L0).

Mount, vary online (if necessary), and attach the future VMREL2 3340 disk to MAINT:

```
vary online raddr
attach cuu to maint as 341
```

Punch a copy of the Format/Allocate program. You must be sure to use a copy of Format/Allocate from the VM/370 system you just updated with the PLC tape because the PLC tape contains the 3340 support. Issue the following commands to punch a copy of the Format/Allocate program:

```
close punch
close reader
purge reader all
access 190 b/a
spool 00d to *
punch ipl fmt (noheader
ipl 00c
```

Next, format, label, and allocate the 3340 disk that is to be the system residence disk. Format the 3340 disk exactly as you would have if you used the 3340 starter system generation procedure.

In the following example, the responses (format, 341, 3340-35, 000, 348 and VMREL2) format the 3340 disk at address 341 and label it VMREL2. The values you should specify for a 3340 Model 70 are shown in parentheses.

```
VM/370 FORMAT/ALLOCATE PROGRAM VERSION 2.0
ENTER FORMAT OR ALLOCATE: format
FORMAT FUNCTION SELECTED
ENTER DEVICE ADDRESS (CCU): 341
ENTER DEVICE TYPE: 3340-35 (or, 3340-70)
ENTER START CYLINDER (XXX) OR "LABEL": 000
ENTER END CYLINDER (XXX): 348 (or, 697)
ENTER DEVICE LABEL: vmrel2
FORMAT STARTED
FORMAT DONE
000 PAGE RECORDS FLAGGED
```

Now that the system residence volume is formatted and labeled, you must allocate the disk space. In the following example, the space on the 3340 disk at address 341, with the label VMREL2, is allocated (11 cylinders of permanent space, 5 cylinders of directory, 310 cylinders of temporary space, and 23 cylinders of TDSK space):

```
ENTER FORMAT OR ALLOCATE: allocate
ALLOCATE FUNCTION SELECTED
ENTER DEVICE ADDRESS (CCU): 341
ENTER DEVICE TYPE: 3340-35 (or, 3340-70)
ENTER DEVICE LABEL: vmrel2
ENTER ALLOCATION DATA FOR VOLUME VMREL2
TYPE CYL CYL
.... ... ...
perm 000 010
drct 011 015
temp 016 325
tdsk 326 348
end
```

```
      ALLOCATION RESULTS
      PERM 000 010
      DRCT 011 015
      TEMP 016 325
      TDSK 326 348
      DEVICE 341 VOLUME VMREL2 ALLOCATION ENDED
```

If your system residence volume is a 3340 Model 70, the allocation results messages show the unused cylinders (349-697) as temporary space.

Next, you must format, label, and allocate a 3340 to contain the required minidisks. This is the volume that is normally labeled CPV2L0. At this time that volume must be labeled something other than CPV2L0 and allocated as permanent space. Later the label will be changed to CPV2L0. In the following example the volume at address 340 is labeled TCPVOL and allocated as permanent space:

```
      ENTER FORMAT OR ALLOCATE: format
      FORMAT FUNCTION SELECTED
      ENTER DEVICE ADDRESS (CCU): 340
      ENTER DEVICE TYPE: 3340-35 (or, 3340-70)
      ENTER START CYLINDER (XXX) OR "LABEL": 000
      ENTER END CYLINDER: 348 (or, 697)
      ENTER DEVICE LABEL: tcpvol
      FORMAT STARTED
      FORMAT DONE
      000 PAGE RECORDS FLAGGED

      ENTER FORMAT OR ALLOCATE: allocate
      ALLOCATION FUNCTION SELECTED
      ENTER DEVICE ADDRESS (CCU): 340
      ENTER DEVICE TYPE: 3340-35 (or, 3340-70)
      ENTER DEVICE LABEL: tcpvol
      ENTER ALLOCATION DATA FOR VOLUME TCPVOL
      TYPE CYL CYL
      .... ... ...
      perm 000 348 (or, perm 000 697)
      end
      ALLOCATION RESULTS
      PERM 000 348
      DEVICE 340 VOLUME TCPVOL ALLOCATION ENDED
      ENTER FORMAT OR ALLCCATE:
```

Enter CP mode and detach the volumes you just formattted and allocated. Then IPL CMS.

```
      detach 340
      detach 341
      ipl 190
```

STEP 6. UPDATE THE CURRENT VM/370 DIRECTORY

Edit your current VM/370 Directory program control statements and add the following control statements to the software support virtual machine (MAINT):

```
      MDISK 290 3340 048 203 TCPVOL MR READ
      MDISK 291 3340 027 014 TCPVOL WR
      MDISK 294 3340 251 098 TCPVOL MR
      MDISK 299 3340 046 002 TCPVOL WR
      MDISK 341 3340 000 348 VMREL2 MW
      MDISK 340 3340 000 001 TCPVOL MW
```

The first five minidisk areas just defined correspond to those defined during the 3340 starter system generation procedure; they are identical except for virtual device addresses and volume labels. The entries just defined are on virtual channel 2; the starter system procedure places the minidisks on virtual channel 1. After the minidisks are moved from the 2314/3330 volume to the 3340 volume, the 3340 minidisk address is changed to virtual channel 1. At this time the virtual device addresses must be different.

The MAINT virtual machine also has the first cylinder of the TCPVOL volume defined so it can change the label to CPV2L0 after the complete 3340 system is built. MAINT has the 3340 VMREL2 volume defined so it can generate a 3340 system residence and load the 3340 VM/370 directory onto it.

After you update the VM/370 Directory program control statements, load the directory on the current system (2314/3330). Issue the CMS DIRECT command to load the directory.

```
direct filename
```

You must log off and log on again to make the new MAINT directory entries effective. Before you log off, attach the 3340 volume (TCPVOL) that is to contain the minidisks to the VM/370 system.

```
attach 340 to system as tcpvol
logoff
```

STEP 7. FORMAT MINIDISK AREAS

Log on again as MAINT, IPL CMS, and format the minidisk areas so that they can be used with CMS.

```
logon maint
ipl 190
```

MAINT now has access to the minidisks on the 2314/3330 volume (addresses 190, 191, 194, and 199) and access to the minidisk areas to be built on the 3340 volume (addresses 290, 291, 294, and 299). You must format 290, 291, 294, and 299. Be careful not to format 190, 191, 194, and 199, as they are the minidisks that must be copied to the 3340 volume.

```
access (nodisk
format 291 a
format 294 a
format 299 a
format 290 a
```

The 290 minidisk eventually becomes the CMS system disk on the TCPVOL 3340 volume. You must invoke FORMAT with the recompute option to make the last two cylinders (which will contain the CMS nucleus) unavailable to the CMS file system.

```
format 290 a 201 (recomp
```

STEP 8. COPY THE MINIDISKS

Copy all the data on MAINT's 190, 191, 194, and 199 minidisks to 290, 291, 294, and 299 minidisks. Issue the following commands:

```
access 290 a
access 190 b
copyfile * * b = = a (replace
```

Repeat the preceding three commands for each minidisk you want copied (191 to 291, 194 to 294, and 199 to 299).

After all the minidisks are copied to the 3340 volume, release the 2314/3330 minidisks, detach the 2314/3330 minidisks, and redefine the virtual addresses. However, do not release or detach the 2314/3330 CMS system disk yet. Issue the following commands:

```
release 191
release 194
release 199
detach 191
detach 194
detach 199
define 291 as 191
define 294 as 194
define 299 as 199
```

At this time all the data is copied onto TCPVOL. The MAINT virtual machine owns the areas on TCPVOL (191, 194, and 199) that it normally would own on CPV2L0 at this point in the 3340 starter system generation procedure. However, the CMS system disk is still on the 2314/3330 volume and 290 is not yet redefined as 190. This change is made when CMS is generated.

STEP 9. UPDATE AND ASSEMBLE THE CP SYSTEM CONTROL (DMKSYS) AND SYSTEM NAME TABLE (DMKSNT) FILES

Access the CP source disk (194).

```
access 194 a
```

Edit the DMKSYS ASSEMBLE file and change the macros to support 3340 disks according to your planned configuration. Also, update the DMKSNT ASSEMBLE file if you wish to save any systems on a 3340 disk. Then assemble the updated files:

```
vmfasm dmksys dmkr20
vmfasm dmksnt dmkr20
```

STEP 10. UPDATE THE VM/370 DIRECTORY AND LOAD IT ON A 3340 VOLUME

Change the VM/370 directory as follows:

- Change the DIRECTORY control statement, specifying 3340 as the device type.
- Replace the current description of the MAINT virtual machine with a copy of the MAINT virtual machine that supports a 3340 system.

Your entry for the MAINT virtual machine should be:

```
USER MAINT CPCMS 16M BCEG
ACCOUNT ACT3 MAINT
OPTION ECMODE REALTIMER
CONSOLE 009 3215
SPOOL 00C 2540 READER A
SPOOL 00D 2540 PUNCH A
SPOOL 00E 1403 A
MDISK 341 3340 000 348 VMREL2 MW
MDISK 190 3340 048 203 CPV2L0 MR READ
MDISK 191 3340 027 014 CPV2L0 MR READ
MDISK 194 3340 251 093 CPV2L0 MR READ
MDISK 199 3340 046 002 CPV2L0 MR READ
```

Now, using the Directory program control statements you just coded, invoke the Directory program and write the new VM/370 directory on the 3340 disk.

```
direct filename
```

where filename is the name of your modified directory.

STEP 11. GENERATE A NEW CP NUCLEUS ON A 3340 VOLUME

The GENERATE EXEC procedure requires a scratch tape mounted and ready at virtual address 182. It writes the CP nucleus on this tape. Invoke the GENERATE EXEC procedure to build and load the new CP nucleus.

```
attach cuu to maint as 182
generate cp nucleus
```

Instructions for using the GENERATE EXEC procedure are in the "Using the GENERATE EXEC Procedure to Generate a New CP or CMS Nucleus" section of Part 6.

STEP 12. GENERATE A NEW CMS NUCLEUS ON A 3340 VOLUME

IPL CMS and invoke the GENERATE EXEC procedure to generate a new CMS nucleus.

```
ipl 190
generate cms nucleus (noload
```

The NOLOAD option indicates that the loadable copy of the CMS nucleus is to be placed in the virtual card reader.

When the GENERATE EXEC procedure completes its processing, enter CP mode, detach both the 2314/3330 and the 3340 CMS system disks, link to the 3340 CMS system disk as 190, and IPL CMS to generate the new CMS nucleus on the 3340 disk (TCPVOL).

```
detach 190
detach 290
link * 290 190
ipl 00c
```

Respond to the following prompting messages as shown:

```
DMSINI606R SYSTEM DISK ADDRESS = 190
DMSINI615R Y-DISK ADDRESS = 19e
DMSINI607R REWRITE THE NUCLEUS? yes
DMSINI608R IPL DEVICE ADDRESS = 190
DMSINI609R NUCLEUS CYL ADDRESS = 201
DMSINI610R ALSO IPL CYLINDER 0? yes
DMSINI611R  VERSION IDENTIFICATION =
DMSINI612R INSTALLATION HEADING =
CMS VERSION 2.0 - mm/dd/yy hh:mm

ipl 190
```

Then generate the CPEREP and ASSEMBLE modules, creating the corresponding auxiliary directories.

```
access 190 a
cmsgend cperep
asmgend
```

STEP 13. CHANGE THE LABEL OF THE 3340 DISK LABELED TCPVOL

Invoke the Format/Allocate program again to change the label of the 3340 disk containing the minidisks from TCPVOL to CPV2L0.

```
VM/370 FORMAT/ALLOCATE PROGRAM VERSION 2.0
ENTER FORMAT or ALLOCATE: format
FORMAT FUNCTION SELECTED
ENTER DEVICE ADDRESS (CCU): 340
ENTER DEVICE TYPE: 3340-35 (or, 3340-70)
ENTER START CYLINDER (XXX) OR "LABEL": label
ENTER DEVICE LABEL: cpv2l0
LABEL IS NOW CPV2L0
```

At this time you have generated CP and CMS on a 3340 disk and copied the minidisks that are needed to support VM/370 to a 3340 disk. The updated CP system control (DMKSYS) and system name table (DMKSNT) files are included in the new CP nucleus and the updated VM/370 is on the 3340 system residence volume.

The 2314/3330 is still intact and updated to the latest PLC level. It is recommended that the VM/370 directory entry for MAINT on the 2314/3330 system be left in its updated form. If you do leave the updates in the 2314/3330 directory, you can build all or part of a 3340 system again.

After all the spool files on the system are processed, shutdown and IPL the new 3340 system residence volume and execute the Installation Verification Procedure (IVP).

At this time the named systems, such as CMS, can be saved.

```
logon maint
ipl 190
savesys cms
```

Appendix I: VM/370 Restrictions

A virtual machine created by VM/370 is capable of running an IBM System/360 or System/370 operating system as long as certain VM/370 restrictions are not violated. Virtual machine restrictions and certain execution characteristics are stated in this chapter.

DYNAMICALLY MODIFIED CHANNEL PROGRAMS

In general, virtual machines may not execute channel programs that are dynamically modified (that is, channel programs that are changed between the time the START I/O (SIO) is issued and the end of the input/output occurs, either by the channel program itself or by the CPU). However, some dynamically modified channel programs are given special handling by CP: specifically, those generated by the Indexed Sequential Access Method (ISAM) running under OS/PCP, OS/MFT, and OS/MVT; those generated by ISAM running in an OS/VS virtual=real partition; and those generated by the OS/VS Telecommunications Access Method (TCAM) Level 5, with the VM/370 option.

The self-modifying channel programs that ISAM generates for some of its operations receive special handling if VM/370 is generated with the ISAM option and if the virtual machine using ISAM has that option specified in its VM/370 directory entry. There is no such restriction for DOS ISAM, or for ISAM if it is running in an OS/VS virtual=virtual partition. If ISAM is to run in an OS/VS virtual=real partition, you must specify the ISAM option in the VM/370 directory entry for the OS/VS virtual machine.

Virtual machines using OS/VS TCAM (Level 5, generated or invoked with the VM/370 option) issue a DIAGNOSE instruction when the channel program is modified. This instruction causes CP to reflect the change in the virtual CCW string to the real CCW string being executed by the channel. CP is then able to execute the dynamically modified channel program properly.

The restriction against dynamically modified channel programs does not apply if the virtual machine has the virtual=real performance option and the NOTRANS option has been set on.

MINIDISK RESTRICTIONS

The following restrictions exist for minidisks:

1. In the case of Read Home Address with the skip bit off, VM/370 modifies the home address data in user storage at the completion of the channel program because the addresses must be converted for minidisks; therefore, the data buffer area may not be dynamically modified during the input/output operation.

2. On a minidisk, if a CCW string uses multitrack search on input/output operations, subsequent operations to that disk must have preceding seeks or continue to use multitrack operations. There is no restriction for dedicated disks.

3. OS/PCP, MFT, and MVT ISAM may be used with a minidisk only if the minidisk is located at the beginning of the physical disk (that is,

at cylinder zero). There is no such restriction for DOS or OS/VS ISAM.

4. VM/370 does not return an end of cylinder condition to a virtual machine that has a virtual 2311 mapped to the top half (that is, tracks 0 through 9) of 2314 or 2319 cylinders.

5. If the user's channel program (CCWs) for a minidisk do not perform a Seek operation, then to prevent accidental accessing, VM/370 inserts a positioning Seek operation into the user's CCWs. Thus, certain channel programs may generate a condition code (CC) of zero on a SIO instead of an expected CC of one, which is reflected to the virtual machine. The final status is reflected to the virtual machine as an interrupt.

6. DASD channel programs directed to minidisks on 3330 or 3340 devices may give different results than on dedicated drives if the channel program includes multiple-track operations and depends on a Search ID High or a Search ID High or Equal to terminate the program. This is because the record 0 count fields on the 3330 and 3340 must contain the real cylinder number of the track on which they reside; therefore, a Search ID High based on a low virtual cylinder number may terminate prematurely if a real record 0 is encountered. This restriction does not apply to minidisks with a relocation factor of zero. This restriction does apply to minidisks with a VTOC greater than one track that are used with OS (Release 20.6 and later) or OS/VS (any release), since the VTOC Locate function uses a Search ID High to stop at the end of the VTOC.

   Note: If the 'R' byte of 'CCHHR' is equal to zero at the time a virtual Start I/O is issued, but the 'CCHHR' field is read in dynamically by the channel program before the SEARCH ID CCW is executed, then the real SEARCH ID CCW uses the relocated 'CCHHR' field instead of the 'CCHHR' field that was dynamically read in. This causes erroneous results. To avoid this problem, the virtual machine should not default the'R' byte of 'CCHHR' to binary zero if the search arguments are to be read in dynamically and a SEARCH ID on Record R0 is not intended.

7. The IBCDASDI program cannot assign alternate tracks for a 3330.

TIMING DEPENDENCIES

Timing dependencies in input/output devices or programming do not function consistently under VM/370:

1. The following telecommunication access methods (or the designated option) violate the restriction on timing dependency by using program-controlled interrupt techniques and/or the restriction on dynamically modified channel programs:

   - OS Basic Telecommunications Access Method (BTAM) with the dynamic buffering option.
   - OS Queued Telecommunications Access Method (QTAM).
   - DOS Queued Telecommunications Access Method (QTAM).
   - OS Telecommunications Access Method (TCAM).
   - OS/VS Telecommunications Access Method (TCAM) Level 4 or earlier, and Level 5 if TCAM is not generated or invoked with the VM/370 option.

These access methods may run in a virtual=real machine with CCW translation suppressed by the SET NOTRANS ON command. (OS BTAM can be generated without dynamic buffering, in which case no virtual machine execution violations occur. However, the BTAM reset poll macro will not execute under VM/370 if issued from third level storage. For example, a reset poll macro has a NOP effect if executed from a virtual=virtual storage under VS1 which is running under VM/370.)

2. Programming that makes use of the PCI channel interrupt for channel program modification or processor signalling must be written so that processing can continue normally if the PCI is not recognized until I/O completion or if the modifications performed are not executed by the channel.

3. Devices that expect a response to an interrupt within a fixed period of time may not function correctly because of execution delays caused by normal VM/370 system processing. An example of such a device is the IBM 1419 Magnetic Character Reader.

4. The operation of a virtual block multiplexer channel is timing dependent. For this reason, the channel appears available to the virtual machine operating system, and channel available interrupts are not observed. However, operations on virtual block-multiplexing devices should use the available features like Rotational Position Sensing to enhance utilization of the real channels.

CPU MODEL-DEPENDENT FUNCTIONS

On the System/370 Model 158 only, the Virtual Machine Assist feature cannot operate concurrently with the 7070/7074 compatibility feature (Feature #7117).

Programs written for CPU model-dependent functions may not execute properly in the virtual machine under VM/370. The following points should be noted:

1. Programs written to examine the machine logout area do not have meaningful data since VM/370 does not reflect the machine logout data to a virtual machine.

2. Programs written to obtain CPU identification (via the Store CPU ID instruction, STIDP) receive the real machine value. When the STIDP instruction is issued by a virtual machine, the version code contains the value 256 in hexadecimal ("FF") to represent a virtual machine.

3. Programs written to obtain channel identification (via the Store Channel ID instruction, STIDC) receive information from the virtual channel block. Only the virtual channel type is reflected; the other fields contain zeroes.

4. No simulation of other CPU models is attempted by VM/370.

VIRTUAL MACHINE CHARACTERISTICS

Other characteristics that exist for a virtual machine under VM/370 are as follows:

1. If the virtual=real option is selected for a virtual machine, input/output operations specifying data transfer into or out of the virtual machine's page zero, or into or out of storage locations whose addresses are greater than the storage allocated by the virtual=real option, must not occur. The storage-protect-key mechanism of the IBM System/370 CPU and channels operates in these situations but is unable to provide predictable protection to other virtual machines. In addition, violation of this restriction may compromise the integrity of the system. The results are unpredictable.

2. VM/370 has no multiple path support and, hence, does not take advantage of the two-channel switch. However, a two-channel switch can be used between the IBM System/370 running a virtual machine under VM/370 and another CPU.

3. The DIAGNOSE instruction cannot be issued by the virtual machine for its normal function. VM/370 uses this instruction to allow the virtual machine to communicate system services requests. The Diagnose interface requires the operand storage addresses passed to it to be real to the virtual machine issuing the DIAGNOSE instruction. For more information about the DIAGNOSE instruction in a virtual machine, see the VM/370: System Programmer's Guide.

4. A control unit normally never appears busy to a virtual machine. An exception exists when a forward space file or backward space file command is executed for a tape drive. Subsequent I/O operations to the same virtual control unit result in a control unit busy condition until the forward space file or backward space file command completes. If the real tape control unit is shared by more than one virtual machine, a control unit busy condition is reflected only to the virtual machine executing the forward space file or backward space file command. When a virtual machine attempts an I/O operation to a device for which its real control unit is busy, the virtual machine is placed in I/O wait (non-dispatchable) until the real control unit is available. If the virtual machine executed a SIOF instruction (rather than SIO) and was enabled for block-multiplexing, it is not placed in I/O wait for the above condition.

5. The number of pages used for input/output must not exceed the total number of user pages available in real storage; violation of this restriction causes the real computing system to be put into an enabled wait state.

6. The CP IPL command cannot simulate self-modifying IPL sequences off dedicated unit record devices or certain self-modifying IPL sequences off tape devices.

7. The VM/370 spooling facilities do not support punch-feed-read, stacker selection, or column binary operations. Detection of carriage control channels is supported for a virtual 3211 only.

8. VM/370 does not support count control on the virtual 1052 operator's console.

9. Programs that use the integrated emulators function only if the real computing system has the appropriate compatibility feature. VM/370 does not attempt simulation. The DOS emulators are not supported.

10. The READ DIRECT and WRITE DIRECT instructions are not supported for a virtual machine.

11. The System/370 SET CLOCK instruction cannot be simulated and, hence, is ignored if issued by a virtual machine. The System/370

STORE CLOCK instruction is a nonprivileged instruction and cannot be trapped by VM/370; it provides the true TOD clock value from the real CPU.

12. The 1050/1052 Model 2 Data Communication System is supported only as a keyboard operator's console. Card reading, paper tape I/O, and other modes of operation are not recognized as unique, and hence may not work properly. This restriction applies only when the 1050 system is used as a virtual machine operator's console. It does not apply when the 1050 system is attached to a virtual machine via a virtual 2701, 2702, or 2703 line.

13. The pseudo-timer (usually device address 0FF, device type TIMER) does not return an interrupt from a Start I/O; therefore, do not use EXCP to read this device.

14. A virtual machine IPL with the NOCLEAR option renders one page of the virtual machine invalid. The IPL simulator uses one page of the virtual machine to initiate the IPL function. The starting address of the invalid page is either the result of the following formula:

$$\frac{\text{virtual machine size}}{2} = \text{starting address of invalid page}$$

or the hexadecimal value 20,000, whichever is smaller.

15. To maintain system integrity, data transfer sequences to and from a virtual system console are limited to a maximum of 2032 bytes. Channel programs containing data transfer sequences that violate this restriction are terminated with an interrupt whose CSW status indicates incorrect length and a channel program check.

    Note: A data transfer sequence is defined as one or more read or write CCWs connected via chain data. The introduction of command chaining defines the start of a new data transfer sequence.

16. If you intend to define more than 73 virtual devices for a single virtual machine, be aware that any single request for free storage in excess of 512 doublewords (a full page) will cause the VM/370 system to abnormally terminate (ABEND code PTR007) if the extra storage is not available on a contiguous page. Therefore, two contiguous pages of free storage must be available in order to log on a virtual machine with more than 73 virtual devices (three contiguous pages for a virtual machine with more than 146 virtual devices, etc.). Contiguous pages of free storage are sure to be available only immediately after IPL, before other virtual machines have logged on. Therefore, a virtual machine with more than 73 devices should be the first to log on after IPL.

17. When an I/O error occurs on a device, the System/370 hardware maintains a contingent connection for that device until a SENSE channel command is executed and sense data is recorded. That is, no other I/O activity can occur on the device during this time. Under VM/370, the contingent connection is maintained until the SENSE command is executed, but I/O activity from other virtual machines can begin on the device while the sense data is being reflected to the virtual machine. Therefore, the user should be aware that on a shared disk, the access mechanism may have moved during this time.

18. The mode setting for 7-track tape devices is maintained by the control unit. Therefore, when a virtual machine issues the SET MODE channel command to a 7-track tape device, it changes the mode setting of all 7-track tape devices attached to that control unit.

This has no effect on virtual machines (such as OS or DOS) that issue SET MODE each time a CCW string is to be executed. However, it can cause a problem if a virtual machine fails to issue a SET mode with each CCW string executed. Another virtual machine may change the mode setting for another device on the same control unit, thereby changing the mode setting of all 7-track tape devices attached to that control unit.

19. OS/VS2 is supported in uniprocessor mode only.

20. For remote 3270s, VM/370 supports a maximum of 16 binary synchronous lines, minus the number of 3704/3705 Communications Controllers in NCP mode minus one (if there are any 3704/3705 Communications Controllers in emulation mode).

21. If an I/O device (such as a disk or tape drive) drops ready status while it is processing virtual I/O activity, any virtual machine users performing I/O on that device are unable to continue processing or to log off. Also, the LOGOFF and FORCE commands are not effective because they do not complete until all outstanding I/O is finished. The system operator should determine which I/O device is involved and make that device ready once more.

CMS RESTRICTIONS

The following restrictions apply to CMS, the conversational subsystem of VM/370:

1. CMS executes only on a virtual IBM System/370 provided by VM/370.

2. The maximum sizes of CMS minidisks are as follows:

Disk	Maximum Cylinders
2314/2319	203
3330 Series	246
3340 Model 35	349
3340 Model 70	682

3. Unit record equipment cannot be dedicated to CMS; the spooling facilities of VM/370 must be used.

4. Only those OS facilities that are simulated by CMS can be used to execute OS programs produced by language processors under CMS.

5. Many types of object programs produced by CMS (and OS) languages can be executed under CMS using CMS's simulation of OS supervisory functions. The following functions, although supported in DOS and OS virtual machines under VM/370, are not supported under CMS:

   - The execution of DOS object programs. Although DOS programs can be assembled under CMS (using the VM/370 Assembler), DOS object programs cannot execute under CMS.
   - The writing or updating of OS data sets and DOS files.

6. CMS can read sequential and partitioned OS data sets and sequential DOS files, by simulating certain OS macros.

   The following restrictions apply when CMS reads OS data sets that reside on OS disks:

- Read-password-protected data sets are not read.
- VSAM, BDAM, and ISAM data sets are not read.
- Multi-volume data sets are read as single-volume data sets. End-of-volume is treated as end-of-file and there is no end-of-volume switching.
- Keys in data sets with keys are ignored and only the data is read.
- User labels in user-labeled data sets are bypassed.

The following restrictions apply when CMS reads DOS files that reside on DOS disks:

- No DOS macros are simulated.
- Only DOS sequential files can be read. CMS options and operands that do not apply to OS sequential data sets (such as the MEMBER and CONCAT options of FILEDEF and the PDS option of MOVEFILE) also do not apply to DOS sequential files.
- The following types of DOS files cannot be read:
  - --DOS VSAM, DAM, and ISAM files.
  - --DOS core image, relocatable, source statement and procedure libraries.
  - --Files with the input security indicator on.
  - --DOS files that contain more than 16 user label and/or data extents. (If the file has user labels, they occupy the first extent; therefore the file must contain no more than 15 data extents.)
- Multi-volume files are read as single-volume files. End-of-volume is treated as end-of-file. There is no end-of-volume switching.
- User labels in user-labeled files are bypassed.
- Since DOS files do not contain BLKSIZE, RECFM, or LRECL parameters, these parameters must be specified via FILEDEF or DCB parameters, otherwise, defaults of BLOCKSIZE=32760 and RECFM=U are assigned. LRECL is not used for RECFM=U files.

MISCELLANEOUS RESTRICTIONS

If you intend to run VM/370 Release 1 and Release 2 systems alternately, apply Release 1 PLC 14 or higher (APAR V1179) to your Release 1 system, to provide compatibility and to prevent loss of spool files in case of a warm start.

Index

A
abbreviations in command lines 349
access modes
  defining for minidisks 145
  of CMS files 32
accessing CMS disks 31
ACCOUNT control statement 141
accounting number, defining in VM/370 directory 141
ACCT option, defining in VM/370 directory 143
Acoustic Coupler feature 84
ADAPTER operand, of RDEVICE macro 101
ADDRESS operand
  of RCHANNEL macro 108
  of RCTLUNIT macro 106
  of RDEVICE macro 98
address
  assigned to system residence volume during system generation procedure 165,186,208
  defining virtual device addresses for RSCS 229
A-disk 26
ALLOCATE control statement of Format/Allocate program 378
allocating
  DASD space for CP use 379
  DASD space on CP-owned volumes 116
  the system residence volume 162,183,204
  3340 volumes when building 3340 system from current system 401
ALTCONS operand of RIOGEN macro 110
Alternate Character Set feature 84
alternate console, defining 110
alternate track assignment, IBCDASDI 372
alternate tracks for minidisks 69
APL
  executing in a virtual machine 42
  restricted use with Network Control Program (NCP) 242
  restricted use with Partitioned Emulation Program (PEP) 243
application programming facilities of VM/370 37
applying
  IBM-supplied updates
    example 311
    with VMFASM 287
  PTFs
    to the 3704/3705 control program 266
    to VM/370 283
  your own updates to VM/370 288
ASMGEND
  generating the Assembler 308
  responses 308
  when to use 347
ASM3705 command 253
  files created by 254
ASP virtual machine 47
  system generation requirements 48
  VM/370 directory requirements 48
Assembler 28
  building the 3705 assembler 246
  for VM/370 335
  optional tape for system generation 160
  updating with VMFBLD 304
  using ASMGEND to generate 308
assembling
  the CP system control file 172,193,215
  the forms control buffer 173,194,215
  the real I/O configuration file 172,193,215
  the RSCS configuration table 330
  the standalone spool remote program 273
  the 3704/3705 macros 253
assigning an alternate track, IBCDASDI 372
Audible Alarm feature 83
Automatic EOB on Carrier Return feature 83
Automatic Ribbon Shift and Line Feed Select feature 83
Automatic Turnaround feature 89
AUX files
  definition 281
  identification records 281
auxiliary directories
  creating for the Assembler 308
  using VMFBLD to create for EREP and the Assembler 304
auxiliary files
  definition 281
  example of 287,289
auxiliary storage, required by CMS 24
available real storage, Formula 1 19
AXSLINKS COPY file 229
  creating 229,234
  GENLINK macro 230

B
back up, using DASD Dump Restore program 363
backing up, the newly generating VM/370 system 177,198,219
BASE control record, ZAP program 326
Base Expansion feature 86
BASEADD operand, of RDEVICE macro 102
Batch Facility 34
  adding routines 34
  EXEC procedures for 34
  VM/370 directory entries 34
BHSET macro 246
BMX option 37
  defining in VM/370 directory 143
BSCGEND, when to use 347
BUILD macro 247
building
  a new CMS nucleus
    creating a CMS system disk 315
    example 314
    generating the CMS nucleus 315
    saving CMS 320

a new CP nucleus, example 311
a new nucleus for CP, CMS, or RSCS 301
a new RSCS nucleus 304,329
creating an RSCS system disk 329
disks needed 329
generating the RSCS nucleus 330
the VM/370 directory during system generation 172,193,214
the 3705 assembler 246

C
calculating
available real storage 19
maximum size of virtual=real area 20
capacity of device when formatted 375
card input, to Format/Allocate program 377
chain link 31
channel index table 110
Channel Indirect Data Addressing feature 75
channel switching 49
between two CPUs 50
for tape 51
on one CPU 51
system generation requirements 50
choices, representation in command formats 350
CHTYPE operand, of RCHANNEL macro 108
CLASS operand
of GENLINK macro 230
of RDEVICE macro 99
CLUSTER macro 246
CMS operand, of VMFBLD command 301
CMS, see Conversational Monitor System (CMS)
CMSGEND 279
example of use 290,321
format 306
generating a CMS module 306
responses 307
when to use 347
COBOL, see Full American National Standard COBOL
commands
equivalent VM/370 and CP-67 commands 384
equivalent VM/370 and CP-67/CMS commands 389
functions of CMS commands 27
notational conventions 349
service program, DDR 354
syntax, comparison of VM/370 and CP-67 384
to load the CMS nucleus 319
ZAP 322
comment control record, ZAP program 328
communications system test virtual machine 44
COMP macro 246
compatibility, of VM/370 and CP-67/CMS 383
concurrent execution of virtual machines 36
configuration aid 337
configuration macros for 3704/3705 251
configuration table, assembling for RSCS 235,330
configurations supported by VM/370 72
CONS operand, of RIOGEN macro 110
CONSOLE control statement 144
console input, to Format/Allocate program 381
console, defining the real system console 110
control card for the loader 296
control files 281
AUX file identification records 281
entering comments 282
example 287,289,309,311
use with VMFLOAD 293
MACS record 281
supplied by IBM 282
DMKR20 CNTRL 282
DMSM20 CNTRL 282
DMSR20 CNTRL 282
DMTR20 CNTRL 282
NCPR20 CNTRL 282
update identification records 281
update level identifier 281
used with VMFLOAD command 292
Control Program (CP) 13
address where modules loaded 296
applying IBM-supplied updates 287
building a new nucleus with VMFBLD 301
commands equivalent to CP-67 commands 384
compatibility with CP-67 383
DASD space requirements 63
disk access 31
error recording, DASD space requirements 64
free storage requirements 61
functions exercised by the IVP 224
generating the nucleus 297
loading the nucleus 175,195,217
loadlist requirements 294
macros, notational conventions 349
minimum configuration 72
nucleus
DASD requirements 63
example of building 311
example of loading 312
loading nucleus with virtual=real area 175,196,217,298
loading nucleus without virtual=real area 175,196,217
reducing the size of 63
special VMFBLD procedure for nucleus with virtual=real area 303
optional tape containing assembly listings 160
paging, DASD requirements 64
real control blocks storage requirements 61
real storage requirements 61
example 62
requirements to regenerate 347
resident nucleus storage requirements 61
saved systems, DASD requirements 64
spooling, DASD requirements 64
system disk, example of copying 365
system support plan 285

trace table storage requirements 61
verifying 224
VM/370 directory, DASD space requirements 64
warm start data, DASD space requirements 64
control statements, DDR service program 355
controlling the operation of the standalone spool remote program 275
CONUNIT operand, of SRP2780 macro 273
Conversational Monitor System (CMS) 13
A-disk 26
applying IBM-supplied updates 287
auxiliary storage requirements 24
Batch Facility 34
building a new nucleus with VMFBLD 303
capacity of virtual disk 30
chain link 31
COBOL execution restrictions 28
command language 27
commands equivalent to CP-67/CMS commands 389
compatibility with CP-67/CMS 383
creating CMS disk-resident modules 321
default device addressess 25
devices supported 25
DIRECT command 137
disk management 29
disk-resident modules 321
disks
access 31
capacity 30
formatting 30
labels 29
linking 31
editing files 34
example
building a new CMS nucleus 314
CMS BATCH 342
virtual machine for 342
executable program products 335
file management 29,30
File Status Table 31
filemodes 32
files sharing 32
files
access modes 32
format 30
maximum number 30
functions exercised by the IVP 224
generating a module 306
generating the nucleus 297
invoking the Directory program 137
libraries 27
limited support of OS and DOS 29
loading during system generation 169,190,211
macros
equivalent to CP-67/CMS macros 397
notational conventions 349
master file directory 30
messages issued while nucleus is being generated 315
minidisk labels 70
minimum configuration 73
modules, when to regenerate 347
nucleus
commands issued to load it 319
example of building 314
generating 315
when it must be regenerated 347
optional tape containing assembly listings 160
planning considerations 24
PL/I execution restrictions 28
printing files 33
program languages available with CMS 28
punching card files 33
reading card files 33
records, maximum number per file 30
requirements to regenerate 347
sample VM/370 directory entry 26
saving a copy of 35
saving during system generation 181,202,223
sharing the system residence volume 40
source programs 320
symbolic names for devices 25
system disk 26,164,185,206
creating 315
moving 178,199,220
system support plan 285
tape support 32
text processing 28
unit record support 26,33
verifying 224
virtual storage requirements 24
COPY statement, DDR service program 357
copying real disks, example 365
Correspondence feature 83
CORTABLE, defining 122
CP operand, of VMFBLD command 301
CP system control file
assembling during system generation 172,193,215
performance considerations 115
preparing 114
printing and punching the starter system supplied copy 171,192,213
reassembling 297
supplied with the starter system 114
SYSCOR macro 122
SYSLOCS macro 125
SYSOPR macro 121
SYSOWN macro 116
SYSRES macro 118
SYSTIME macro 123
updating and assembling when building 3340 system from current system 404
CP, see Control Program (CP)
CPNAME operand
of NAMENCP macro 155
of RDEVICE macro 102
CPSIZE operand, of NAMENCP macro 155
CPT-TWX
ICA features 87
sign-on procedure 266
2701 required features 85
2702 required features 86
2703 features 86
CPTYPE operand
of NAMENCP macro 155
of RDEVICE macro 101

CPUs
  desirable features 75
  required features 75
    for RSCS 91
  supported by VM/370 75
CP-67, commands equivalent in VM/370 384
CP-67/CMS
  commands equivalent in VM/370 389
  compatibility with VM/370 383
  macros equivalent in VM/370 397
creating a 3340 system
  allocating 3340 volumes 401
  applying PLC tape 400
  changing the label of the 3340 disk 406
  copying minidisks 404
  formatting minidisks 403
  formatting 3340 volumes 401
  from a current system 399
  generating a new CMS nucleus 405
  generating a new CP nucleus 405
  labeling 3340 volumes 401
  loading
    CMS 399
    new CP macro library 399
    the new VM/370 directory 404
  updating and assembling
    CP system control file 404
    the system name table 404
  updating
    the current VM/370 directory 402
    the real I/O configuration file 400
  what you need 399
creating an RSCS system disk 329
current system, using to generate a 3340 system 399
CUTYPE operand, of RCTLUNIT macro 106
cylinder zero format, 3330 376

D
DADEF statement, IBCDASDI 368
DASD Dump Restore program 354
  control statements 355
  COPY statement 357
  copying real disks 365
  copying the CMS system disk 365
  creating a standalone punch file 297
  creating a standalone version on disk 297
  DUMP statement 357
  function statements 357
  INPUT statement 355
  invoking as a standalone program 354
  invoking under CMS 354
  I/O definition statements 355
  loading from the starter system tape 163,184,205
  OUTPUT statement 355
  PRINT statement 359
  RESTORE statement 357
  SYSPRINT statement 357
  TYPE statement 359
  using to back up disks 363
  using to back up system residence volume 364
  using to back up the newly generating VM/370 177,198,219
  using to restore starter system to disk 163,184,205
DASD space
  allocating for the VM/370 directory 135
  allocating on CP-owned volumes 116
  calculating cylinders needed to contain warm start data 120
  for CMS minidisks 65
  formatting for the VM/370 directory 135
  needed by CMS 24
  required by CP 63
  required by CP nucleus 63
  reserving for a CP nucleus with a virtual=real area 18
  reserving for 3704/3705 control program image 59
DASD
  configuration aid for 338
  control units supported by VM/370 77
  required features 78
  restrictions 78
  supported by VM/370 77
Data Set Attachment and Dialup feature 82
data transmissicn paths, defining in your RSCS system 229
DATETIME macro 246
DDR (see DASD Dump Restore program)
DDR command 354
DEDICATE control statement 147
  example 147
dedicated lines, restricted use with Partitioned Emulation Program (PEP) 243
defaults, representation in command formats 350
defective tracks on minidisks 69
defining
  data transmission paths for RSCS 229
  minidisks 66
  total number of tag slots for RSCS 229
  virtual device addresses for RSCS 229
device addresses, changing for the loader 295
devices
  channel switching 49
  coding the RDEVICE macro for system console 98
  coding the RDEVICE macro for unsupported devices 98
  configuration aid for 337
  CPUs supported by VM/370 75
  DASD supported by VM/370 77
  default addresses for CMS 25
  defining for starter system 165,186,207
  defining the subclass for unsupported devices 100
  linking at logon 148
  magnetic tape supported by VM/370 79
  making available during system generation procedure 168,189,211
  needed for system generation 159
  real devices supported by VM/370 72
  remote spooling devices supported by VM/370 89
  sample configuration 111
  simulated by programming 149
  supported by CMS 25

terminals supported by VM/370 81
unit record devices supported by VM/370 80
unsupported devices dedicated to virtual machines 92
DEVTYPE operand, of RDEVICE macro 98
DIAGNOSE X'50' instruction 261
DIAL command
restricted use with Network Control Program (NCP) 242
restricted use with Partitioned Emulation Program (PEP) 243
DIALSET macro 246
direct access storage device (see DASD)
DIRECT command 137
format 138
DIRECT operand, of GENERATE command 299
DIRECTORY control statement 139
Directory program 136
ACCOUNT control statement 141
CONSOLE control statement 144
control statements 138
creating a standalone punch file 297
creating a standalone version on disk 297
DEDICATE control statement 147
DIRECTORY control statement 139
invoking under CMS 137
IPL control statement 143
LINK control statement 148
MDISK control statement 144
OPTION control statement 141
running standalone 138
SPECIAL control statement 149
SPOOL control statement 146
USER control statement 139
disks
channel switching between two CPUs 50
channel switching on one CPU 51
CMS, access 31
formatting for CMS 30
labels, CMS 29
management under CMS 29
display devices, configuration aid for 338
distribution code, defining in the VM/370 directory 141
DMKFCB (see forms control buffer)
DMKFCB operand, of GENERATE command 299
DMKRIO operand, of GENERATE command 299
DMKRIO, sample file 113
DMKR20 CNTRL 282
DMKSNT (see system name table)
DMKSNT operand, of GENERATE command 300
DMKSRP (see standalone spool remote program)
DMKSYS operand, of GENERATE command 299
DMKSYS, supplied with the starter system 114
DMSM20 CNTRL 282
DMSR20 CNTRL 282
DMTLOC macro library 229
creating 234
DMTR20 CNTRL 282
DOS
Initialize Disk utility program 69
initializing minidisks for 68
support under CMS 29
DUMP control records, ZAP program 324
DUMP statement, DDR service program 357
dumps, example of virtual machine that receives system dumps 341
Dynamic Translation Facility (DAT) 383

E
EBCDIC Code feature 85
EBCDIC feature 83
EBCDIC Transmission Code feature 89
EBCDIC Transparency feature 85,89
ECMODE option 37
defining in VM/370 directory 142
EDIT macro 246
editing, CMS files 34
Editor 34
EIA Interface with Clock feature 83
Emulation Program (EP) 239
see also 3704/3705 control program
how to code the RDEVICE macro 56
macro coding considerations 252
special considerations for loading 263
support under VM/370 241
emulators, integrated emulators under VM/370 336
enabling, the line for the standalone spool remote program 274
END control record, ZAP program 328
END statement, IBCDASDI 371
ENDBH macro 246
EP, see Emulation Program (EP)
EREP, updating with VMFBLD 304
error recording cylinders, defining 119
error recording, DASD requirements 64
example
applying
IBM-supplied updates 311
your own updates to VM/370 289
auxiliary files 287,289
backing up
minidisks with DDR program 363
system residence volume with DDR program 364
building
a new CMS nucleus 314
a new CP nucleus 311
control file 287,289,309,311
copying
real disks 365
the CMS system disk 365
files used to update VM/370 310
loading an updated CP nucleus 312
MTA macros 250
saving CMS 320
SRP2780 macro 273
update files 287,289,311
using CMSGEND 290,321
using control file with VMFLOAD 293
using VMFASM
to update modules 287
to update source files 311
using VMFLOAD 290,292
3704/3705 configuration macros 251
EXEC procedure
to help you install the standalone spool remote program 275
used to regenerate CMS 347

executing VMFBLD during system generation procedure 170,191,212
execution, concurrent virtual machines 36
Expanded Capability feature 85
Expansion feature 85
extended floating-point feature 76

F
FEATURE operand
  of RCTLUNIT macro 106
  of RDEVICE macro 99
features
  CPU
    Channel Indirect Data Addressing 75
    extended floating-point 76
    floating-point 75
    system timing facility 75
    virtual machine assist feature 75
    Word Buffer 75,76
  required by DASD 78
  RSCS 89
  TCUs
    Base Expansion 86
    EBCDIC Code 85
    EBCDIC Transparency 85
    Expanded Capability 85
    Expansion 85
    Synchronous Data Adapter Type II 85
    Telegraph Adapter Type II 85
    31 Line Expansion 86
  terminals
    Acoustic Coupler 84
    Alternate Character Set 84
    Audible Alarm 83
    Automatic EOB on Carrier Return 83
    Automatic Ribbon Shift and Line Feed Select 83
    Data Set Attachment and Dialup feature 82
    EBCDIC or Correspondence 83
    EIA Interface with Clock 83
    for the 2741 81
    Half-duplexed 83
    IBM Line Adapter 82
    Lower Case Character Display 83
    Operator Identification Card Reader 83
    Pin Feed Platen 82
    Print Inhibit 82
    PTTC/EBCD 81
    Receive Interrupt 82
    Red Ribbon Control 82
    Transmit Interrupt 82
    Typamatic Keys 82
    1050 features 82
    1051 control unit features 82
    1200 bps Integrated Modem/Interrupt 83
    2741 START/STOP 83
    3270 83
  Two-channel Switch 92
  2780
    Automatic Turnaround 89
    EBCDIC Transmission Code 89
    EBCDIC Transparency 89
    144 Character Print Line feature 89
    3704/3705, Multiple Terminal Access (MTA) 249
file directory 30
file management, CMS 29,30
File Status Table 31
filemodes 32
files
  CMS
    editing 34
    maximum number of records 30
    printing 33
    punching 33
    reading 33
  maximum number per CMS disk 30
  sharing in CMS 32
floating-point feature 75
FORMAT (CP) 69
FORMAT control statement, Format/Allocate program 378
format
  of CMS files 30
  of commands, interpreting 350
  of READ control statement 173,194,215
  of 3330 cylinders for use by CP 375
FORMAT, use with CMS minidisks 68
Format/Allocate program 377
  card input 377
  console input 381
  creating a standalone punch file 297
  creating a standalone version on disk 297
  loading from starter system tape 161,182,203
formatted, device capacity when 375
formatting
  CMS disks 30
  minidisks during system generation 178,199,221
  minidisks to contain CMS source programs 320
  minidisks when building 3340 system from current system 403
  the A-disk for the standalone spool remote program 272
  the RSCS system disk 233
  the system residence volume 161,182,203
  volumes, general information 375
  3340 volumes when building 3340 system from current system 401
forms control buffer
  assembling during system generation 173,194,215
  printing and punching the starter system supplied copy 171,192,213
  reassembling 297
  supplied with starter system 156
Formula 1 (calculating available real storage) 19
Formula 2 (calculating maximum size of virtual=real area) 20
FORTRAN Interactive Debug 335
FORTRAN IV 28
free storage, permanently allocated for CP 61
Full American National Standard COBOL 28
  execution restrictions under CMS 28
function statements, DDR service program 357

G
GENERATE 279,297
  DIRECT operand 299
  DMKFCB operand 299
  DMKRIO operand 299
  DMKSNT operand 300
  DMKSYS operand 299
  format 298
  invoking 173,194,216
  IPLDECK operand 300
  NUCLEUS operand 300
  SRVCPGM operand 300
  using during system generation 171,191,213
  using to load CP nucleus 175,195,217
  virtual addresses assumed during processing 297
  VM370 operand 298
  when to use 348
generating
  a CMS nucleus 297,315
    messages issued 315
  a CP nucleus 297
  a new nucleus 292
  a new RSCS nucleus 330
    assembling the configuration table 330
    copying supervisor files to the system disk 330
    copying the line drivers to the system disk 331
    loading the RSCS files 330
  a 3340 system from a current system 399
  RSCS
    assembling the configuration table 235
    building the RSCS nucleus 235
    creating the COPY files that define your RSCS system 233
    creating the DMTLOC macro library 234
    formatting the RSCS system disk 233
    loading 236
    loading CMS 232
    loading the PLC tape 232
  the 3704/3705 control program 237,244
    applying PTFs 266
    ASM3705 command 253
    assembling the 3705 source files 257
    building the text library 257
    building the 3705 assembler 246
    coding the BUILD macro 247
    coding the HOST macro 249
    coding the macros 246
    coding the SYSCNTRL macro 248
    configuration macros 251
    defining the macro and text libraries 252
    defining the necessary files 257
    GEN3705 command 255
    GROUP macro 251
    link editing the 3705 text files 257
    LKED command 258
    LOADCC EXEC file 245
    loading it 262
    loading the program from tape 245
    logging on 244,264
    required macro libraries 246
    required text libraries 246
    SAVENCP command 261
    saving it 260
    setting up the CMS virtual machine 244
    Stage 1 252
    Stage 2 255
  3340 system residence volume from a current system 15
GENLINE macro
  format 231
  LINE operand 231
GENLINK macro 47
  CLASS operand 230
  format 230
  ID operand 230
  KEEP operand 230
  LINE operand 230
  requirements for coding 230
  TASK operand 230
  TYPE operand 230
GENTAGQ macro
  format 231
  NUM operand 231
GEN3705 command 255
  file created 256
GETALT statement, IBCDASDI 373
graphic devices, eliminating support for 63
GROUP macro 251

H
Half-duplexed feature 83
hardware service, example of virtual machine for 342
hardware support, virtual machines 127
HOST macro 249

I
IBCDASDI
  assigning alternate tracks for minidisks 69
  creating a standalone punch file 297
  DADEF statement 368
  END statement 371
  general information 366
  GETALT statement 373
  invoking 374
  IPLTXT statement 371
  JOB statement 368
  LASTCARD statement 371
  MSG statement 368
  restrictions 366
  statements 367
  use with OS and DOS minidisks 68
  VLD statement 370
  VTOCD statement 370
IBM Line Adapter feature 82
IBM program numbers, for program products 335
IBM program products
  executable under CMS 335
  IBM program numbers 335
  integrated emulators 336
ICA (see Integrated Communications Attachment)

ID operand
of GENLINK macro 230
of SYSTIME macro 124
IEHDASDI 69
IEHDASDR 69
initialization
with surface analysis 366
without surface analysis 367
initializing a minidisk 366
INPP (DOS Initialize Disk utility program) 69
input control records, ZAP 323
INPUT statement, DDR service program 355
input
card, to Format/Allocate program 377
console, to Format/Allocate program 381
Installation Verification Procedure (IVP) 15,224
CMS functions tested 224
CP functions tested 224
executing
in a single virtual machine 226
with virtual machines other than IVPM1 and IVPM2 226
without testing system abnormal termination processing 226
interpreting, the results 227
starting it 225
virtual machine requirements 224
Installed User Programs (IUPs)
MUSIC 335
SCRIPT/370 335
supported by VM/370 335
installing
CP and CMS 14
RSCS 15
the standalone spool remote program (DMKSRP) 272
Integrated Communications Attachment, features 87
interactive virtual machines 36
internal pointers, generating 125
interpreting the results of IVP processing 227
interruption codes, for the loader 296
introduction, to the 3704/3705 control program 239
invoking
DDR as a standalone program 354
DDR under CMS 354
IBCDASDI 374
the GENERATE EXEC procedure 173,194,216
VMFBLD
during RSCS system generation 232
to build the RSCS nucleus 235
I/O definition statements, DDR service program 355
I/O waits in a virtual machine 39
IOLIST macro 246
IPL control statement 143
IPL, CP nucleus with virtual=real area 19
IPLDECK operand, of GENERATE command 300
IPLTXT statement, IBCDASDI 371
IPLUNIT operand, of SPR2780 macro 272
ISAM option 37
defining in VM/370 directory 142
ISAM, channel programs 38

J
JOB statement, IBCDASDI 368

K
KEEP operand, of GENLINK macro 230

L
LABEL function, Format/Allocate program 380
label
CMS disk labels 29
for minidisks 69
of starter system volume 163,184,205
labeling
DASD volumes 380
the starter system volume 163,183,205
the system residence volume 162,183,204
3340 volumes when building 3340 system from current system 401
LASTCARD statement, IBCDASDI 371
LAXLINES COPY file 229
creating 231,234
GENLINE macro 231
libraries, macro libraries for CMS 27
LINE macro 251
LINE operand
of GENLINE macro 231
of GENLINK macro 230
LINELIST macro 246
LINK command 31
sharing minidisks 70
LINK control statement 148
link editing the 3705 files 258
link, definition 229
linkage editor control statements 258
linking, CMS disks 31
LKED command 258
files created 259
load map 296
LOADCC
loading the 3704/3705 control program from tape 245
when to use 348
loader 295
changing device addresses 295
interruption codes 296
load map 296
set page boundary (SPB) control cards 294
wait conditions 295
loading
newly generated VM/370 system 176,197,218
the standalone spool remote program 274
the 3704/3705 control program 262
loadlist
file 292
for CP nucleus with virtual=real area 293
for CP nucleus without virtual=real area 293
pageable modules for CP 294
requirements for CP 294
LOC operand, of SYSTIME macro 123

locked pages 45
logging on
  after NCP abnormal termination 266
  through lines controlled by the 3704/3705 control program 264
logical records, CMS 30
Lower Case Character Display feature 83

M
macro libraries
  creating the DMTLOC macro library for RSCS 234
  defining for 3704/3705 system generation 252
  generating for RSCS 229
  NCPMAC01 246
  NCPMAC02 246
  required to generate the 3704/3705 control program 246
  updating with VMFMAC 291
  used to create 3340 system from a current system 399
macros
  BHSET 246
  BUILD 247
  CLUSTER 246
  COMP 246
  DATETIME 246
  DIALSET 246
  EDIT 246
  ENDBH 246
  equivalent VM/370 and CP-67/CMS macros 397
  GENLINE 231
  GENLINK 47
  GENTAGQ 231
  GROUP 251
  HOST 249
  IOLIST 246
  LINE 251
  LINKLIST 246
  MTALCST 249
  MTALIST 249
  MTATABL 249
  NAMENCP 155
  NAMESYS 153
  notational conventions 349
  OS macro libraries under CMS 27
  RCHANNEL 108
  RCTLUNIT 106
  RDEVICE, to support 3704/3705 control program 54
  RIOGEN 110
  SERVICE 246
  simulation of OS macros under CMS 29
  SRP2780 272
  STARTBH 246
  SYSCNTRL 248
  SYSCOR 122
  SYSLOCS 125
  SYSOPR 121
  SYSOWN 116
  SYSRES 118
  SYSTIME 123
  TERMINAL 251
  TSO macro libraries under CMS 27
  UBHR 246
MACS record, of control files 281
magnetic tape
  control units supported by VM/370 79
  devices supported by VM/370 79
master file directory 30
MAXDIAL operand, of RDEVICE macro 102
maximum size of virtual=real area 20
MDISK control statement 144
  example 146
MDISK statements, sharing minidisks 70
messages, issued while CMS nucleus is being generated 315
minidisks 66
  see also virtual disks
  backing up with DASD Dump Restore program 363
  copying when building 3340 system from current system 404
  defining
    in VM/370 directory 144
    primary access mode 145
  example of use and definition 67
  formatting during system generation 178,199,221
  initializing 366
    for OS and DOS 68
  labels 69
  linking to 70
  MDISK control statement 70
  multi-write access of 71
  overlapping extents 127
  passwords for 71
  permanent 66
  read-only access of 71
  read/write access of 71
  sharing among virtual machines 70
  temporary 66
  track characteristics 68
minimum configuration
  for CMS 73
  for RSCS 73
  for VM/370 72
minimum truncations in command lines 349
MNOTE messages
  for RCHANNEL macro 109
  for RCTLUNIT macro 107
  for SYSCOR macro 122
  for SYSOWN macro 117
  for SYSRES macro 120
  for SYSTIME macro 124
  for the RDEVICE macro 102
model numbers, of the 3704/3705 55
MODEL operand, of RDEVICE macro 99
modules
  CP, address where loaded 296
  creating disk-resident CMS modules 321
  generating a CMS module 306
  updating with VMFASM 286
  when to regenerate for CMS 347
moving, the CMS system disk 178,199,220
MSG statement, IBCDASDI 368
MTALCST macro 249
MTALIST macro 249
MTATABL macro 249
Multiple Terminal Access (MTA) feature 249
  sign-on procedure 265
multiple virtual machines using the same operating system 39

multiple-access virtual machines 40
  communications system test 44
  performance considerations 45
  virtual transmission control units 45
multiprogramming operating systems under VM/370 38
multi-write access of minidisks 71
MUSIC (McGill University System for Interactive Computing) 335

N
NAME control record, ZAP program 325
NAMENCP macro
  CPNAME operand 155
  CPSIZE operand 155
  CPTYPE operand 155
  format 155
  SYSPGCT operand 155
  SYSSTRT operand 155
  SYSVOL operand 155
NAMESYS macro
  example 154
  format 153
  SYSCYL operand 153
  SYSHRSG operand 154
  SYSNAME operand 153
  SYSPGCT operand 154
  SYSPGNM operand 154
  SYSSIZE operand 153
  SYSSTRT operand 153
  SYSVOL operand 153
  VSYSADR operand 153
  VSYSRES operand 153
NCP, see Network Control Program (NCP)
NCPDUMP service program 267
NCPMAC01 macro library 246
NCPMAC02 macro library 246
NCPR20 CNTRL 282
Network Control Program (NCP) 239
  see also 3704/3705 control program
  how to code, the RDEVICE macro 56
  logging on after abnormal termination 266
  sign-on procedure 265
  special considerations for loading 263
  support under VM/370 242
NETWORK LOAD command 262
  execution of 262
NUCLEUS operand, of GENERATE command 300
nucleus
  building the RSCS nucleus 235
  CMS
    building a new one 314
    generating 297
    when it must be regenerated 347
  CP
    DASD requirements 63
    example of building 311
    generating 297
    loading 175,195,217
    loading one with virtual=real area 298
  end of resident nucleus address 296
  generating 292
  real storage requirements for CP 61
  reducing the size for CP 63
  RSCS
    building 304,329
    generating 330
NUM operand, of GENTAGQ macro 231
numbers, of IBM program products 335

O
OBJ3705 text library 246
online message, logging on 3704/3705 lines 264
operating systems
  core-image copy saved on disk 60
  multiprogramming under VM/370 38
  naming 153
  options
    BMX 37
    ECMODE 37
    ISAM 37
    REALTIMER 37
  performance guidelines 16
  planning considerations 36
  restrictions for executing under VM/370 36
  sharing the system residence volume 39
  spooling considerations 39
  that execute under VM/370 14
  using reserve/release 52
  virtual machine assist feature 23
operations facilities of VM/370 37
Operator Identification Card Reader feature 83
operator, example of VM/370 directory entry 341
OPTION control statement 141
options, for virtual machines 37
OS COBOL Interactive Debug 335
OS Code & Go FORTRAN 335
OS FORTRAN IV (G1) 335
OS FORTRAN IV (H) Extended 335
OS FORTRAN Library (Mod II) 335
OS FORTRAN OV Library (Mod I) 335
OS Full American National Standard COBOL Version 4 Library 335
OS PL/I Checkout Compiler 335
OS PL/I Optimizing Compiler 335
OS PL/I Optimizing Compiler and Libraries 335
OS PL/I Resident Library 335
OS PL/I Transient Library 335
OS
  initializing minidisks for 68
  ISAM channel programs 38
  macro libraries for CMS 27
  support under CMS 29
OS/VS COBOL Compiler and Library 335
OS/VS COBOL Library Only 335
output spooling classes, defining with the RDEVICE macro 100
OUTPUT statement, DDR service program 355
overlapping extents on minidisks 127

P
PAGE operand, of SYSOWN macro 117
page waits, in a virtual machine 38
pageable modules for CP 294

paging
DASD requirements 64
performance considerations 115
Partitioned Emulation Program (PEP) 239
see also 3704/3705 control program
how to code, the RDEVICE macro 57
macro coding considerations 252
sign-on procedure for lines in NCP mode 265
special considerations for loading 263
support under VM/370 242
passwords, for minidisks 71
PEP, see Partitioned Emulation Program (PEP)
performance guidelines 16
for heavy I/O production 115
for installation with large number of CMS users 115
for I/O waits 39
for multiple-access virtual machines 45
for page waits 38
for read-only minidisks 115
multiprogramming systems 39
virtual machine assist feature 23
when coding the DMKSYS macros 115
physical records, CMS 30
Pin Feed Platen feature 82
planning considerations
for a virtual=real machine 17
for CMS 24
for supporting 3704/3705 53
for the 3704/3705 control program 240
operating systems other than CMS 36
RSCS 46
Planning Systems Generator/CMS 335
planning, for system generation 11
PLC tape
applying to CMS during system generation 179,200,222
applying while building a 3340 system 400
loading during RSCS generation procedure 232
loading during system generation 169,190,212
RSCS contents 228
used to install RSCS 228
PL/I 28
execution restrictions under CMS 28
Print Inhibit feature 82
PRINT statement
DDR service program 359
sample output 362
printing, files in CMS 33
program languages, available with CMS 28
programmable terminals, required features for RSCS 90
programming
application 37
facilities of VM/370 36
programs, service, VM/370 353
PRTRDR operand, of SRP2780 macro 273
PRTUNIT operand, of SRP2780 macro 273
PTFs
applying to the 3704/3705 control program 266
applying to VM/370 283
PTTC/EBCD feature 81
punching
card files in CMS 33
service programs during system generation 171,191,213
the DMKSRP TEXT file 274
the starter system supplied CP system control file 171,192,213
the starter system supplied forms control buffer 171,192,213
the starter system supplied system name table 171,192,213
the starter system supplied VM/370 directory 171,192,213
PUNRDR operand, of SRP2780 macro 273
PUNUNIT operand, of SRP2780 macro 273

R
RCHANNEL macro
ADDRESS operand 108
CHTYPE operand 108
example 108
format 108
MNOTE messages 109
RCHBLOK
see also real control blocks
creating 108
label 108
RCTLUNIT macro
ADDRESS operand 106
configuration aid for 337
CUTYPE operand 106
example 107
FEATURE operand 106
FEATURE=xxx-DEVICE operand 105
format 106
MNOTE messages 107
RCUBLOK
see also real control blocks
addressing 105
creating 105
label 105
RDEVBLOK
see also real control blocks
creating 97
label 97
RDEVICE macro
ADAPTER operand 101
ADDRESS operand 98
BASEADD operand 102
CLASS operand 99
coding
for system console 98
for unsupported devices 98
to support the 3704/3705 control program 54
configuration aid for 337
CPNAME operand 102
CPTYPE operand 101
defining
output spooling classes 100
subclass for unsupported devices 100
DEVTYPE operand 98
example 102
3704/3705 58
FEATURE operand 99
format 97

how to code
to support the EP 3704/3705 control program 56
to support the NCP 3704/3705 control program 56
to support the PEP 3704/3705 control program 57
MAXDIAL operand 102
MNOTE messages 102
MODEL operand 99
SETADDR operand 101
summary of coding considerations for 3704/3705 control program 57
unit record error messages 103
3704/3705 error messages 103
READ control statement, format of 173,194,215
reading, card files in CMS 33
read-only access of minidisks 71
read/write access of minidisks 71
real configuration, sample 111
real control blocks
defining 95
real storage requirements 61
real control units
DASD control units supported by VM/370 77
magnetic tape control units supported by VM/370 79
unit record control units supported by VM/370 80
3270 control units supported by VM/370 83
real devices
coding the RDEVICE macro
for the system console 98
for unsupported devices 98
configuring 337
CPUs supported by VM/370 75
DASD supported by VM/370 77
dedicating to virtual machines 147
magnetic tape supported by VM/370 79
needed for system generation 159
remote spooling devices supported by VM/370 89
restriction for system console during system generation 161,182,203
sample configuration 111
supported by VM/370 72
supported only in virtual machines 92
terminals supported by VM/370 81
unit record devices supported by VM/370 80
real I/O configuration file
assembling during system generation 172,193,215
example of VM/370 system 111
preparing 96
RCHANNEL macro 108
RCTLUNIT macro 105
RDEVICE macro 97
reassembling 297
RIOGEN macro 110
updating, when creating a 3340 system from a current system 400
real storage
calculating maximum size virtual=real area 19
requirements for VM/370 61
REALTIMER option 37
defining in VM/370 directory 141
Receive Interrupt feature 82
records, maximum number per CMS file 30
Red Ribbon Control feature 82
regenerating
CMS modules 347
service programs 348
Remote Spooling Communications Subsystem (RSCS) 13
applying IBM-supplied updates 287
assembling the configuration table 330
building a new nucleus 304,329
configuration table 229
assembling 235
copying
line drivers to the system disk 331
supervisor files to the system disk 330
CPU required features 91
creating the COPY files that define it 233
defining
all data transmission paths 229
an RSCS virtual machine 229
more than one RSCS virtual machine 47
total number of tag slots 229
virtual device addresses 229,231
disks needed to build a new nucleus 329
distributed on PLC tape 228
example of use 46
example of virtual machine for 342
generating and installing 228
generating the macro library 229
generation procedure 231
assembling the configuration table 235
creating the DMTLOC macro library 234
defining your system 229
formatting the RSCS system disk 233
loading CMS 232
loading RSCS 236
loading the PLC tape 232
link
defining 229
definition 229
loading the files on disk 304
macros, notational conventions 349
minimum configuration 73
nucleus
building 235
generating 330
planning considerations 46
programmable terminals required features 90
required features 89
sample VM/370 directory entry 229
system disk
copying line drivers to 331
copying supervisor files to 330
creating 329

system support plan 285
virtual machine requirements 229
what you need before you generate RSCS 232
2770 required features 89
2780 required features 90
3770 required features 90
3780 required features 90
REMUNIT operand, of SRP2780 macro 273
REP control record, ZAP program 328
requirements
for changing the VM/370 directory 137
RSCS, to run multiple concurrent RSCS virtual machines 229
to regenerate CP and CMS 347
reserved pages 46
reserve/release 52
resident nucleus, ending address 296
resource ID, recording at end of Stage 1 255
RESTORE statement, DDR service program 357
restoring, starter system to disk 163,184,205
restrictions
for operating systems under VM/370 36
IBCDASDI 366
Network Control Program (NCP)
DIAL command 242
sign-on procedure 242
TERMINAL APL command 242
Partitioned Emulation Program (PEP)
dedicated lines 243
DIAL command 243
TERMINAL APL command 243
RIOGEN macro
ALTCONS operand 110
CONS operand 110
example 110
format 110
RMSIZE operand, of SYSCOR macro 122
RSCS operand, of VMFBLD command 301

S
saved systems 60
CMS 35
DASD requirements 64
defining 153
saving CMS 320
saving CMS during system generation 181,202,223
3704/3705 control program 260
SAVENCP command 261
SAVENCP command 261
execution of 261
SCRIPT/370 28,335
SERVICE macro 246
service programs 322
creating a standalone punch file 297
creating a standalone version on disk 297
Directory program 136
loader 295
NCPDUMP 267
punching during system generation 171,191,213
updating 280
VM/370 353
DASD Dump Restore (DDR) 354
FORMAT 377
when they must be regenerated 348
ZAP 322
set page boundary (SPB) control cards 294,296
SETADDR operand, of RDEVICE macro 101
sign-on procedure for 3704/3705 lines in NCP mode 242,265
software support, virtual machines 128
source files, updating 311
special considerations
for loading the 3704/3705 control program 262
for user with only one tape drive 173,194,215
SPECIAL control statement 149
SPOOL control statement 146
spooling classes, defining with the RDEVICE macro 100
spooling
considerations 39
DASD requirements 64
defining virtual devices for 146
performance considerations 115
SRP2780 macro 272
CONUNIT operand 273
example 273
IPLUNIT operand 272
PRTRDR operand 273
PRTUNIT operand 273
PUNRDR operand 273
PUNUNIT operand 273
REMUNIT operand 273
SRVCPGM operand, of GENERATE command 300
Stage 1 generation procedure for 3704/3705 252
recording resource IDs at end 255
Stage 2 generation procedure for 3704/3705 255
special considerations 257
standalone spool remote program 269
controlling the operation of 275
enabling the line 274
installing 272
assembling with VMFASM 273
creating the DMKSRP ASSEMBLE file 272
enabling the line 274
EXEC procedure to aid you 275
formatting the A-disk 272
loading the program 274
logging on the 2780 virtual machine 273
punching the DMKSRP TEXT file 274
introduction 271
loading 274
SRP2780 macro 272
testing 275
STARTBH macro 246
starter system volume
contents of restored disk 164,185,206
format after system generation 177,198,219
format of 2314 restored disk 164
format of 3330 restored disk 185
format of 3340 restored disk 206

IPL of 165,186,207
labeling 163,183,205
required label 163,184,205
starter system
see also 2314 starter system
see also 3330 starter system
see also 3340 starter system
defining devices for 165,186,207
starting the IVP 225
statements, IBCDASDI 367
storage, for each model of the 3704/3705 55
support packages, for the 3704/3705 control program 240
SVCOFF option, defining in VM/370 directory 143
symbolic names for CMS devices 25
Synchronous Data Adapter Type II feature 85
SYSCNTRL macro 248
SYSCOR macro
example 122
format 122
MNOTE messages 122
RMSIZE operand 122
SYSCYL operand, of NAMESYS macro 153
SYSDUMP operand, of SYSOPR macro 121
SYSERR operand, of SYSRES macro 119
SYSHRSG operand, of NAMESYS macro 154
SYSLOCS macro
example 125
format 125
SYSNAME operand, of NAMESYS macro 153
SYSNUC operand, of SYSRES macro 118
SYSOPER operand, of SYSOPR macro 121
SYSOPR macro
example 121
format 121
SYSDUMP operand 121
SYSOPER operand 121
SYSOWN macro
example 117
format 116
MNOTE messages 117
PAGE operand 117
TEMP operand 117
volid operand 117
SYSPGCT operand
of NAMENCP macro 155
of NAMESYS macro 154
SYSPGNM operand, of NAMESYS macro 154
SYSPRINT statement, DDR service program 357
SYSRES macro
example 120
format 118
MNOTE messages 120
special coding considerations 118
SYSERR operand 119
SYSNUC operand 118
SYSRES operand 118
SYSTYPE operand 118
SYSVOL operand 118
SYSWRM operand 119
SYSRES operand, of SYSRES macro 118
SYSSIZE operand, of NAMESYS macro 153
SYSSTRT operand
of NAMENCP macro 155
of NAMESYS macro 153
system abnormal termination, testing with the IVP 225
system console
coding the RDEVICE macro for 98
configuration aid for 337
defining 110
restriction during system generation 161,182,203
system disk 26,29
creating for CMS 315
creating for RSCS 329
example of copying 365
system generation 159
applying PLC changes to CMS 179,200,222
assembling the CP system control file 172,193,215
assembling the forms control buffer 173,194,215
assembling the real I/O configuration file 172,193,215
basic tapes 160
building the VM/370 directory 172,193,214
defining your system 95
general information 160
introduction 14
optional tapes 160
options
to improve performance 16
virtual=real 17
planning considerations 14
planning for 11
planning for RSCS installation 46
procedure for generating RSCS 231
publications to have available 15
RDEVICE macro, coding considerations for 3704/3705 57
requirements
for ASP virtual machine 48
to support channel switching between two CPUs 50
to support channel switching on one CPU 51
restriction on system console address 161,182,203
setting the terminal mode 167,188,209
special considerations for users with only one tape drive 173,194,215
special procedure for generating a virtual=real area 174,195,216
starter systems 14
2314 starter system 161
3330 starter system 182
3340 starter system 203
3704/3705 requirements 53
system name table
creating an entry for 3704/3705 59
for saved systems 60
NAMENCP macro 155
NAMESYS macro 153
preparing 151
printing and punching the starter system supplied copy 171,192,213
reassembling 297
supplied with starter system 151

updating and assembling when building 3340 system from current system 404
system operator, defining 121
system programming facilities of VM/370 36
system residence volume
allocating 162,183,204
assigning address for system generation procedure 165,186,208
backing it up 177,198,219,364
defining or attaching for system generation procedure 167,188,210
formatting 161,182,203
labeling 162,183,204
loading the newly generated VM/370 176,197,218
sharing among multiple virtual machines 39
system support
plan 285
virtual machines 127
system timing facility 75
system, defining during system generation 95
SYSTIME macro
example 124
format 123
ID operand 124
LOC operand 123
MNOTE messages 124
ZONE operand 123
SYSTYPE operand, of SYSRES macro 118
SYSVOL operand
of NAMENCP macro 155
of NAMESYS macro 153
of SYSRES macro 118
SYSWRM operand, of SYSRES macro 119

T
tag slots
defining the total number 229
specifying the total number 231
TAGQUEUE COPY file 229
creating 231,234
GENTAGQ macro 231
tape
basic tapes for system generation 160
channel switching 51
configuration aid for 338
loading the 3704/3705 control program from the distribution tape 245
optional tapes for system generation 160
special considerations for users with only one tape drive 173,194,215
special considerations for 800 bpi tape during system generation 165,185,207
starter system
loading the DASD Dump Restore program 163,184,205
loading the Format/Allocate program 161,182,203
support for CMS 32
TASK operand, of GENLINK macro 230
TCU (see transmission control units)
Telegraph Adapter Type II feature 85
teleprocessing units, under control of multiple-access virtual machine 41
TEMP operand, of SYSOWN macro 117
TERMINAL APL command
restricted use
with Network Control Program (NCP) 242
with Partitioned Emulation Program (PEP) 243
TERMINAL macro 251
terminal mode, setting during system generation procedure 167,188,209
terminals
desirable features 82
required features and special considerations 81
supported as virtual system consoles 81
supported by the 3704/3705 control program under VM/370 239
supported by VM/370 81
system generation restriction 81
under control of multiple-access virtual machine 40
1050
desirable features 83
features 82
1051 control unit features 82
3270 features 83
testing
the operation of the standalone spool remote program 275
the 3704/3705 control program 267
text libraries
defining for 3704/3705 system generation 252
OBJ3705 246
text processing under CMS 28
Time-of-Day (TOD) clock, defining 123
trace table, CP real storage requirements 61
tracks
alternate 69
assigning with IBCDASDI 372
characteristics for minidisks 68
handling defective tracks for minidisks 69
transmission control units
configuration aid for 337
Integrated Communications Attachment features 87
supported by VM/370 85
2701 required features 85
2702 required features 85
2703 features 86
3704/3705 features 87
Transmit Interrupt feature 82
truncation of commands 349
TSO, macro libraries for CMS 27
Two-channel Switch feature 92
Typamatic Keys features 82
TYPE operand, of GENLINK macro 230
TYPE statement
DDR service program 359
sample output 362

U
UBHR macro 246
unit record devices
configuration aid for 339
support for CMS 26

unit record
  control units supported by VM/370 80
  devices supported by VM/370 80
  error messages for the RDEVICE macro 103
  support for CMS 33
unsupported devices 98
  defining the subclass for 100
UPDATE command 279
  UPDATES file 290
  UPDLOG file 290
update files, example of 287,289,311
update identification records 281
update level identifier 281
update procedures, for modules 280
UPDATES file 290
updating
  a module 280
  CMS modules 306
  macro libraries 291
  modules with VMFASM 286
  the Assembler 308
  VM/370 277
    auxiliary files 281
    CMSGEND 279
    control files 281
    disks accessed 309
    example of virtual machine for 341
    files needed 310
    GENERATE 279
    recommended procedures 309
    requirements to regenerate CP and CMS 347
    suggested virtual machine 283
    the service programs 280
    UPDATE command 279
    VMFASM 279
    VMFLOAD 279
    VMFMAC 279
    when to regenerate the service programs 348
UPDLOG file 290
USER control statement 139
using the DDR service program 354
using the Format/Allocate program 377

V
VERIFY control record, ZAP program 327
Verifying CP and CMS 224
VFMASM, example of use 287
VFMLOAD, format 292
VIRT=REAL option, defining in VM/370 directory 142
virtual device addresses, defining for RSCS 231
virtual devices
  teleprocessing units 41
  terminals 45
  transmission control units 45
  3270 terminals 40
virtual disks
  see also minidisks
  capacity under CMS 30
  CMS, maximum number of files 30
  defining in VM/370 directory 144
  management under CMS 29
virtual machine assist feature 23,75
  restrictions 23
  support 23
Virtual Machine Facility/370 (VM/370)
  application programming facilities 37
  applying
    IBM-supplied updates 287
    your own updates 288
  Assembler 335
  channel switching 49
  commands eqivalent to CP-67 commands 384
  commands equivalent to CP-67/CMS commands 389
  compatibility with CP-67/CMS 383
  components of 13
  concurrent virtual machines 36
  DASD space requirements for VM/370 directory 64
  direct access storage devices supported 77
  disks accessed during updating 309
  Emulation Program (EP) support 241
  example of virtual machine for testing 342
  files for a system update 310
  Installed User Programs (IUPs) 335
  interactive virtual machines 36
  introduction 13
    to system generation 14
  macros equivalent to CP-67/CMS macros 397
  multiprogramming systems 38
  Network Control Program (NCP) support 242
  notational conventions for commands and macros 349
  operating system restrictions 36
  operating systems that execute under VM/370 14
  operations facilities 37
  Partitioned Emulation Program (PEP) support 242
  real storage requirements 61
  recommended procedures for updating 309
  service programs 353
  support of 3704/3705 control program 241
  system definition 95
    CP system control file 114
    forms control buffer load 156
    real I/O configuration file 95
    system name table 151
    VM/370 directory 126
  system programming facilities 36
  system support plan 285
  terminals supported 81
  transmission control units supported 85
  Two-channel switch feature 92
  unsupported devices in virtual machines 92
  updating 277
    auxiliary files 281
    CMSGEND 279
    control files 281
    GENERATE 279
    recommended procedures 309
    suggested virtual machine 283

the service programs 280
UPDATE command 279
VMFASM 279
VMFLOAD 279
VMFMAC 279
with VMFASM 287
virtual machines
ACCT option 143
application programming 37
automatic loading at logon 143
BMX option 143
concurrent execution 36
dedicating real devices to 147
defining 126
accounting number 141
distribution code for 141
in the VM/370 directory 139
one for use with RSCS 229
spooling devices 146
the virtual console in the VM/370 directory 144
ECMODE option 142
example
APL 42
ASP 48
CMS 342
CMS BATCH 342
communications system test 44
machine that receives system dumps 341
multiple-access operating system 40
one for hardware service 342
operator's virtual machine 341
RSCS 47
user who updates and supports VM/370 341
virtual TCU controlling 3270 terminals 45
generation procedure for virtual=real 174,195,216
hardware support 127
interactive execution 36
I/O waits 39
ISAM option 142
linking devices at logon 148
maximum number of devices 127
multiple-access 40
multiprogramming operating systems 38
naming systems 153
operating systems 13
options 37
BMX 37
ECMODE 37
ISAM 37
locked pages 45
REALTIMER 37
reserved pages 46
page waits 38
REALTIMER option 141
RSCS 342
saving copy of operating system 60
sharing minidisks 70
software support 128
spooling considerations 39
SVCOFF option 143
system support 127
testing VM/370 342
using reserve/release 52
VIRT=REAL option 142
virtual storage
required by CMS 24
required to IPL CP system with virtual=real area 18
virtual system console, supported by VM/70 81
virtual=real area
creating 174,195,216
loading a CP nucleus that has one 175,196,217,298
special procedure when using VMFBLD 303
specifying the size 174,195,216
virtual=real
calculating maximum size of area 19
examples 20
determining the size of the virtual=real area 19
function 16
generating 17
generation procedure 174,195,216
maximum size 20
reserving DASD space for CP nucleus 18
restrictions 17
specifying for a virtual machine 16
using the option effectively 17
virtual storage requirements 18
VLD statement, IBCDASDI 370
VMFASM 279
example of use 289,311
files created 290
format 286
updating modules 280
using to apply IBM-supplied updates 287
using to assemble RSCS configuration table 330
using to assemble the standalone spool remote program 273
using to update modules 286
VMFBLD
building
a new nucleus 301
the CMS nucleus 303
the CP nucleus 301
the RSCS nucleus 304
CMS operand 301
CP operand 301
executing during system generation 170,191,212
format 301
invoking during RSCS generation procedure 232
invoking to build RSCS nucleus 235
RSCS operand 301
special procedure for CP nucleus with virtual=real area 303
updating EREP and the Assembler and creates corresponding auxiliary directories 304
using to apply PLC changes to CMS 180,201,222
VMFLOAD 279
example of use 290,292
using to generate a new nucleus 292

VMFMAC 279
  format 291
  updating macro libraries 291
VM/370 directory
  allocating DASD space for 135
  building
    a new one 297
    during system generation 172,193,214
  CMS Batch requirements 34
  considerations for preparing entries 126
  control statements for creating 138
  DASD requirements 64
  defining volume to contain 139
  formatting DASD space for 135
  loading it when building 3340 system from current system 404
  maximum number of devices per virtual machine 127
  MDISK statements 70
  options
    BMX 37
    ECMODE 37
    ISAM 37
    REALTIMER 37
  printing and punching the starter system supplied copy 171,192,213
  requirements
    for ASP virtual machine 48
    for changing 137
  RSCS requirements 46
  sample entry 341
    APL virtual machine 43
    APLSYS 345
    BATCH 344
    CMS 26,37
    CMS BATCH 342
    hardware service 342
    MAINT 343
    OPERATNS 343
    OPERATOR 343
    OS2 345
    RSCS 229,342,344
    SRMAINT 343
    testing VM/370 342
    TESTSYS 344
    the machine that receives dumps 341
    the operator 341
    the user who updates and supports VM/370 341
    updating 2314 system 283
    updating 3330 system 283
    updating 3340 system 284
    USER1 USER2 and USER3 344
    2780 344
    2780 virtual machine 274
  supplied with the starter system 129
  updating when building 3340 system from current system 402
VM370 operand, of GENERATE command 298
VM/370
  see Virtual Machine Facility/370 (VM/370)
  minimum configuration 72
  real devices supported 72
  update procedures 279
volid operand, of SYSOWN macro 117
Volume Table of Contents (VTOC) 68
VS BASIC 28,335
VS BASIC Processor 335
VSYSADR operand, of NAMESYS macro 153
VSYSRES operand, of NAMESYS macro 153
VTOCD statement, IBCDASDI 370

W
waits, for the loader 295
warm start data
  calculating the number of cylinders needed 120
  DASD requirements 64
  defining cylinders for 119
Word Buffer feature 75,76,78

X
xxx-DEVICE feature of the RCTLUNIT macro 105

Z
ZAP
  command 322
  control record
    BASE 326
    comment 328
    DUMP 324
    END 328
    NAME 325
    REP 328
    VERIFY 327
  input control records 323
  option output 322
  service program 322
    special considerations for 329
ZONE operand, of SYSTIME macro 123

1
1050
  features 82
  ICA features 87
  MTA sign-on procedure 265
  2701 required features 85
  2702 required features 85
  2703 features 86
1051, features 82
1200 bps Integrated Modem/Interupt feature 83
144 Character Print Line feature 89

2
2305
  allocating DASD space for VM/370 directory 135
  DASD space requirements for
    CP 63
    CP nucleus 63
    error recording 64
    paging and spooling 64
    saved systems 64

VM/370 directory 64
warm start data 64
2314 starter system
defining or attaching the system residence volume 167
DMKSYS file supplied 114
printing and punching 171
forms control buffer supplied 156
printing and punching 171
introduction 14
loading
CMS 169
DASD Dump Restore program 163
Format/Allocate program 161
PLC tape 169
making devices available 168
preparing
the starter system volume 162
the system residence volume 161
restoring to disk 163
setting the terminal mode 167
system generation procedure 161
system name table supplied 151
printing and punching 171
VM/370 directory supplied 130
printing and punching 171
2314
allocating DASD space for VM/370 directory 135
alternate tracks for minidisks 69
capacity for CMS minidisks 65
DASD space requirements for
CP 63
CP nucleus 63
CP nucleus with virtual=real area 18
error recording 64
paging and spooling 64
saved systems 64
VM/370 directory 64
warm start data 64
suggested virtual machine for updating 2314 system 283
2319
allocating DASD space for VM/370 directory 135
alternate tracks for minidisks 69
2701, required features 85
2702, required features 85
2703, required features 86
2741 START/STOP feature 83
2741
features 81
ICA features 87
MTA sign-on procedure 265
2701 required features 85
2702 required features 85
2703 features 86
2770
ICA features 87
required features for RSCS 89
2701 required features 85
2703 features 86
2780 Spool Remote Program, required features 89
2780
see also standalone spool remote program
controlling the operation of 275
ICA features 87
required features for RSCS 90
required features for the 2780 Spool Remote Program 89
terminal characteristics 271
testing 275
2701 required features 85
2703 features 86
3
31 Line Expansion feature 86
3270
coding the RDEVICE macro for the operator identification reader 99
control units 83
features 83
3330 starter system
defining or attaching the system residence volume 188
DMKSYS file supplied 114
printing and punching 192
forms control buffer supplied 156
printing and punching 192
introduction 14
loading
CMS 190
DASD Dump Restore program 184
Format/Allocate program 182
PLC tape 190
making devices available 189
preparing
the starter system volume 183
the system residence volume 182
restoring to disk 184
setting the terminal mode 188
system generation procedure 182
system name table supplied 151
printing and punching 192
VM/370 directory supplied 131
printing and punching 192
3330
allocating DASD space for VM/370 directory 135
alternate tracks for minidisks 69
capacity for CMS minidisks 65
cylinder zero format 376
DASD space requirements for
CP 63
CP nucleus 63
CP nucleus with virtual=real area 18
error recording 64
paging and spooling 64
saved systems 64
VM/370 directory 64
warm start data 64
suggested virtual machine for updating 3330 system 283
system generation restriction 78
3330/IBCDASDI, restrictions 366
3340 starter system
defining or attaching the system residence volume 210
DMKSYS file supplied 114
printing and punching 213
forms control buffer supplied 156
printing and punching 213
introduction 14

loading
CMS 211
DASD Dump Restore program 205
Format/Allocate program 203
PLC tape 212
making devices available 211
preparing
the starter system volume 204
the system residence volume 203
restoring to disk 205
setting the terminal mode 209
system generation procedure 203
system name table supplied 151
printing and punching 213
using alternate procedure to generate a 3340 system 399
VM/370 directory supplied 133
printing and punching 213
3340
allocating DASD space for VM/370 directory 135
alternate tracks for minidisks 69
capacity for CMS minidisks 65
DASD space requirements for
CP 63
CP nucleus 63
CP nucleus with virtual=real area 18
error recording 64
paging and spooling 64
saved systems 64
VM/370 directory 64
warm start data 64
generating a 3340 system from a current system 399
generating 3340 system residence from a current system 15
suggested virtual machine for updating 3340 system 284
3704 and 3705 support packages 240
documentation supplied 240
machine readable material 241
3704/3705 control program
coding the RDEVICE macro to support 54
Emulation Program (EP) 239
generating 237
generating VM/370 to support it 53
generation procedure 244
applying PTFs 266
ASM3705 command 253
assembling the 3705 source files 257
building the text library 257
building the 3705 assembler 246
coding the BUILD macro 247
coding the HOST macro 249
coding the macros 246
coding the MTA macros 249
coding the SYSCNTRL macro 248
configuration macros 251
defining the macro and text libraries 252
defining the necessary files 257
example of configuration macros 251
example of MTA macros 250
GENEND macro 251
GEN3705 command 255
GROUP macro 251
link editing the 3705 text files 257
LKED command 258
LOADCC EXEC file 245
loading 262
loading the program from tape 245
logging on 264
logging on VM/370 244
macro coding consideration for PEP 252
macro coding considerations for EP 252
recording the resource IDs 255
required macro libraries 246
required text library 246
SAVENCP command 261
saving it 260
Stage 1 252
Stage 2 255
introduction 239
link editing 258
loading 262
macros
assembling 253
BHSET 246
BUILD 247
CLUSTER 246
coding considerations for PEP and EP 252
COMP 246
DATETIME 246
DIALSET 246
EDIT 246
ENDBH 246
GENEND 251
GROUP 251
GROUP 251
HOST 249
IOLIST 246
LINE 251
LINELIST 246
SERVICE 246
STARTBH 246
SYSCNTRL 248
TERMINAL 251
to support the Multiple Terminal Access (MTA) feature 249
UBHR 246
minimum storage required 239
NCPDUMP service program 267
Network Control Program (NCP) 239
NETWORK LOAD command 262
Partitioned Emulation Program (PEP) 239
planning considerations 240
related publications 240
setting up a CMS virtual machine 244
support packages 240
support under VM/370 241
terminals supported 239
testing 267
3704/3705
coding considerations for RDEVICE macro 57
creating an entry in the system name table 59
eliminating CP support for 63
error messages for the RDEVICE macro 103
example of RDEVICE macro 58
features not supported 88
models supported 55

naming the control program 155
reserving DASD space for image of control program 59
storage sizes 55
supported models 87
what they support 241
3705 assembler
building 246
invoking 253
3767
features 83
ICA features 87
3770
ICA features 87
required features for RSCS 90
2701 required features 85
2703 features 86
3780, required features for RSCS 90

IBM Technical Newsletter

This Newsletter No. GN20-2658
Date March 31, 1975

Base Publication No. GC20-1801-4
File No. S370-34 (VM/370 Release 2 PLC 13)

Previous Newsletters None

IBM Virtual Machine Facility/370: Planning and System Generation Guide

© IBM Corp. 1975

This Technical Newsletter, a part of Release 2 PLC 13 of IBM Virtual Machine Facility/370, provides replacement pages for your publication. These replacement pages remain in effect for subsequent VM/370 releases unless specifically altered. Pages to be removed and/or inserted are listed below.

Title Page,2	61-64.1	175,176
Preface	67,68	191-192.1
Summary of Amendments	71,72	197,198
	81-86	211-214.1
Contents	89,90	217,218
Figures	95-96.8	233-236
15-18	97-106	239-248.1
23-26	141-142.1	295-296.1
33,34	145-150	337-340
37-40.2	169,170	406.1-406.7
52.1-52.8		

Changes or additions to the text and illustrations are indicated by a vertical line to the left of the change.

SUMMARY OF AMENDMENTS

This Technical Newsletter incorporates changes reflecting support for:

- Remote 3270s
- VM/VS Handshaking
- VM/370 Measurement Facility

Note: Please file this cover letter at the back of your publication to provide a record of changes.

IBM Corporation, VM/370 Publications, 24 New England Executive Park, Burlington, Massachusetts 01803

Printed in U.S.A.

READER'S COMMENTS

Title: IBM Virtual Machine Facility/370: Planning and System Generation Guide

Order No. GC20-1801-4

Please check or fill in the items; adding explanations/comments in the space provided.

Which of the following terms best describes your job?

☐ Customer Engineer	☐ Manager	☐ Programmer	☐ Systems Analyst
☐ Engineer	☐ Mathematician	☐ Sales Representative	☐ Systems Engineer
☐ Instructor	☐ Operator	☐ Student/Trainee	☐ Other (explain below)

How did you use this publication?

☐ Introductory text ☐ Reference manual ☐ Student/ ☐ Instructor text

☐ Other (explain) _____

Did you find the material easy to read and understand? ☐ Yes ☐ No (explain below)

Did you find the material organized for convenient use? ☐ Yes ☐ No (explain below)

Specific criticisms (explain below)

Clarifications on pages _____

Additions on pages _____

Deletions on pages _____

Errors on pages _____

Explanations and other comments:

Trim Along This Line

Thank you for your cooperation. No postage necessary if mailed in the U.S.A.

GC20-1801-4

YOUR COMMENTS PLEASE . . .

This manual is one of a series which serves as a reference source for systems analysts, programmers, and operators of IBM systems. Your comments on the back of this form will be carefully reviewed by the persons responsible for writing and publishing this material. All comments and suggestions become the property of IBM.

Please note: Requests for copies of publications and for assistance in utilizing your IBM system should be directed to your IBM representative or to the IBM sales office serving your locality.

FOLD FOLD

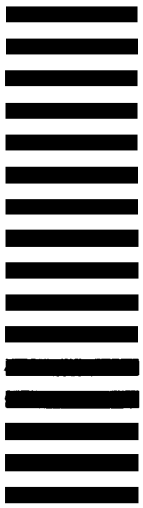
FIRST CLASS
PERMIT NO. 172
BURLINGTON, MASS.

BUSINESS REPLY MAIL

NO POSTAGE STAMP NECESSARY IF MAILED IN U.S.A.

POSTAGE WILL BE PAID BY

IBM CORPORATION
VM/370 PUBLICATIONS
24 NEW ENGLAND EXECUTIVE PARK
BURLINGTON, MASS. 01803



FOLD FOLD

IBM

International Business Machines Corporation
Data Processing Division
1133 Westchester Avenue, White Plains, New York 10604
(U.S.A. only)

IBM World Trade Corporation
821 United Nations Plaza, New York, New York 10017
(International)

Trim Along This Line

IBM VM/370: Planning and Sys Gen Guide

Printed in U.S.A.

GC20-1801-4