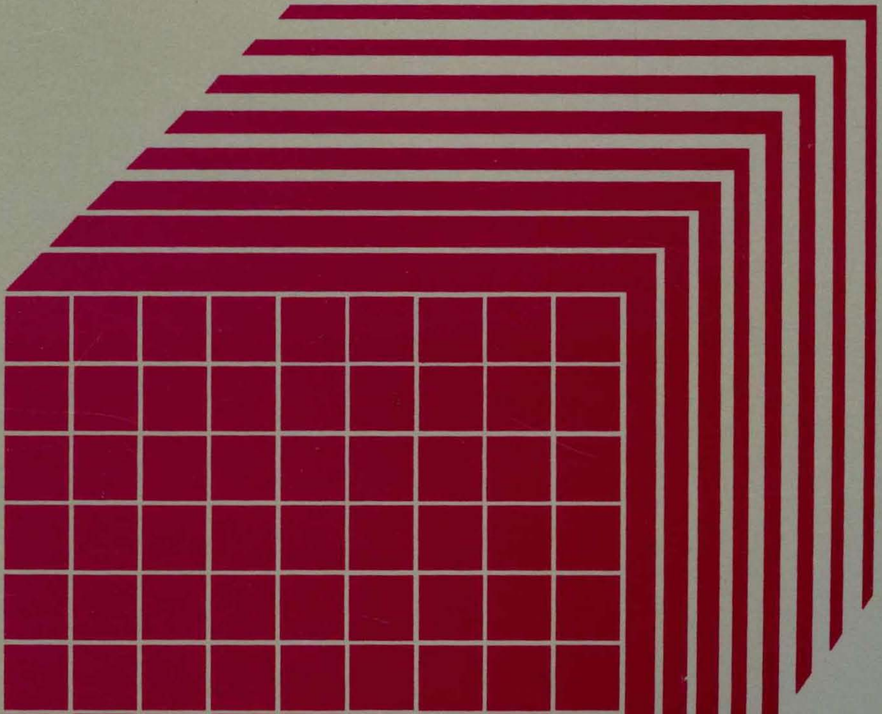
IBM

Virtual Machine/
System Product

CMS User's Guide

Release 5

SC19-6210-4



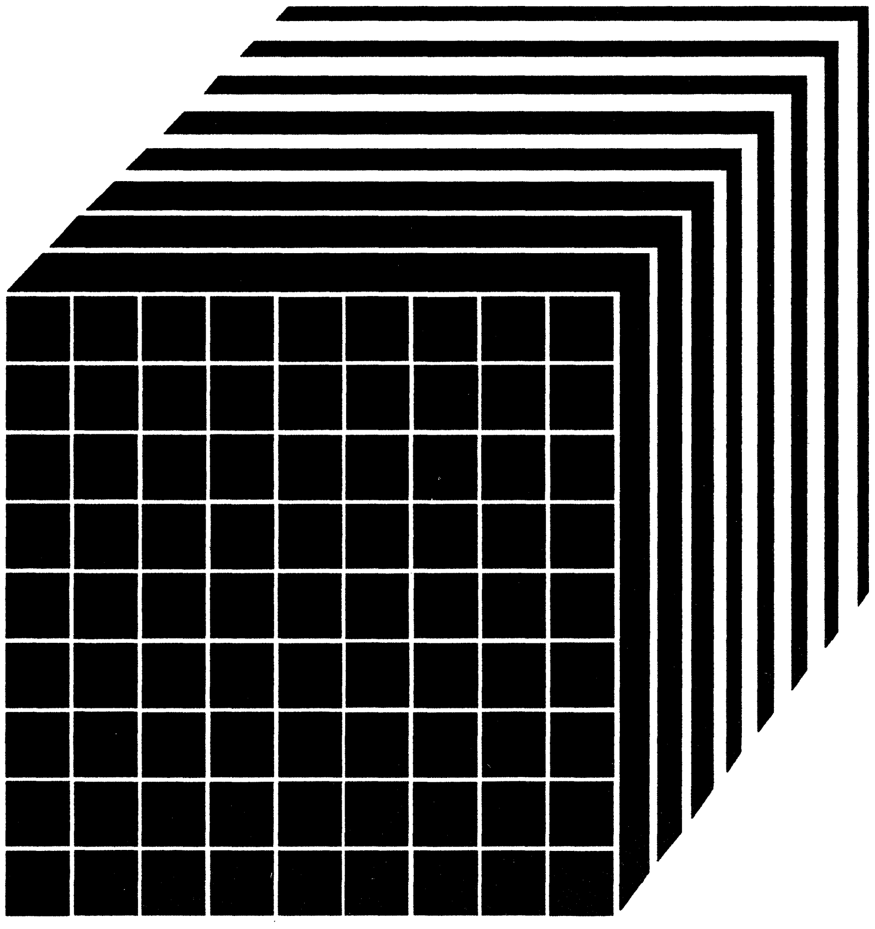
IBM

Virtual Machine/ System Product

CMS User's Guide

Release 5

SC19-6210-4



Fifth Edition (December 1986)

This edition, SC19-6210-4, is a major revision of SC19-6210-3 and applies to Release 5 of the IBM Virtual Machine/System Product, (VM/SP), program number 5664-167, and to all subsequent releases and modifications until otherwise indicated in new editions or Technical Newsletters. Changes are periodically made to the information contained herein; before using this publication in connection with the operation of IBM systems, consult the *IBM System/370, 30XX, and 4300 Processors Bibliography*, GC20-0001, for the editions that are applicable and current.

Summary of Changes

For a list of changes, see "Summary of Changes" on page 337.

Changes or additions to the text and illustrations are indicated by a vertical line to the left of the change.

In this manual are illustrations in which names are used. These names are fanciful and fictitious; they are used solely for illustrative purposes and not for identification of any persons or company.

Ordering Publications

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

A form for readers' comments is provided at the back of this publication. If the form has been removed, comments may be addressed to IBM Corporation, Information Development, Dept. G60, P.O. Box 6, Endicott, NY, U.S.A. 13760. IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

The form for readers' comments provided at the back of this publication may also be used to comment on the VM/SP online HELP facility.

Preface

This publication is intended for the general VM/SP user. It contains information describing the interactive facilities of VM/SP and includes examples showing you how to use VM/SP.

Since this book has been restructured for Release 5, system programmers or users interested in specific programming environments running under CMS should now refer to *VM/SP CMS for System Programming*, SC24-5286.

This publication contains four parts, plus appendixes.

"Part 1: Getting Acquainted with VM/SP" contains sections that describe, in general terms, the CMS facilities and the CP and CMS commands that you can use to control your virtual machine. If you are an experienced programmer who has used interactive terminal systems before, you may be able to refer directly to the *VM/SP CMS Command Reference* to find specific details about CMS commands that are summarized in this part. Otherwise, you may need to refer to later sections of this publication to gain a broader background in using CMS.

The topics discussed in Part 1 are:

- Introduction to VM/SP
- VM/SP Environments and Mode Switching
- The CMS File System
- What You Can Do with CMS Commands
- Using the HELP Facility.

"Part 2: Working with VM/SP" discusses the procedures to be followed for performing routine CMS tasks.

The topics discussed in Part 2 are:

- Editing Your Files
- Using Real Printers, Punches, Readers and Tapes
- Communicating with Other Computer Users
- Looking at VM/SP Through Windows
- Using the CMS Batch Facility.

"Part 3: Learning To Use EXECs" gives detailed information on creating EXEC procedures to use with CMS.

The topics discussed in Part 3 are:

- Introduction to the EXEC Processors
- Creating System Product Interpreter EXECs
- Creating a PROFILE EXEC
- CMS Commands Used With System Product Interpreter EXECs.

"Part 4: Tailoring Your System" contains information on how to use the various functions of VM/SP to customize your virtual machine.

The topics discussed in Part 4 are:

- Customizing Full-Screen CMS
- Tailoring the HELP Facility.

"Appendix A: Considerations for Line-mode Terminals" discusses aspects of VM/SP and CMS that are different or unique when you use these terminals.

"Appendix B: Summary of CMS Commands" lists the commands available in the CMS command environment.

"Appendix C: Summary of CP Commands" describes the CP command privilege classes and summarizes the commands available in the CP command environment.

"Appendix D: Considerations for Full-Screen CMS and Windowing" provides a basic overview of windowing and full-screen CMS functions and gives some specific information regarding CP, CMS, System Product Editor, and application interactions with windowing and full-screen CMS.

"Glossary of Terms and Abbreviations" lists and defines terms that are used in this manual.

Contents

Part 1: Getting Acquainted with VM/SP 1

Chapter 1. Introduction to VM/SP 3
How You Communicate With VM/SP 3
  What You Must Know to Use VM/SP 6
Beginning Your Terminal Session 6
  Logging On at a 3270-Type Terminal 7
  Logon Exceptions 8
  Getting Into CMS 9
Ending Your Terminal Session 10
Entering Commands 10
RETRIEVE Function 11
Setting Program Function Keys 12
Using Full-Screen CMS 13
Display Screen Characteristics 14
  Messages 14
  VM Status Notices 14
  Full-Screen CMS Status Notices 17
  Additional Display Screen Capabilities 18
  How VM/SP Responds to Your Commands 19
Working with CMS 21
Virtual Disks and How They Are Defined 23
  Defining Temporary Virtual Disks 24
  Formatting Virtual Disks 24
Sharing Virtual Disks: Linking 25
Identifying Your Disk To CMS: Accessing 26
  Releasing Virtual Disks 27
  Console Output 27

Chapter 2. VM/SP Environments and Mode Switching 31
The CP Environment 33
The CMS Environment 34
  Protected Application Environment 35
  XEDIT and CMS Subset 35
  DEBUG 37
  CMS/DOS 38
Interrupting Program Execution 38
  Using the Attention Key 39
  Interrupting Your Programs 39
  Virtual Machine Interruptions 40
  Control Program Interruptions 41
  Address Stops and Breakpoints 42
Using APL 42
  Error Situations 43
  Leaving the APL Environment 43
Using the 3270 Text Feature 44

Error Situations 44

Chapter 3. The CMS File System 47
CMS File Formats 47
How CMS Files Get Their Names 47
Duplicate Filenames or Filetypes 48
What Are Reserved Filetypes? 50
Filetypes for CMS Commands 50
Output Files: TEXT and LISTING 55
Filetypes for Temporary Files 56
Filetypes for Documentation 57
Filemode Letters and Numbers 57
When to Specify Filemode Letters: Reading Files 60
When to Specify Filemode Letters: Writing Files 61
How Filemode Numbers are Used 62
When To Enter Filemode Numbers 64
Managing Your CMS Disks 65
CMS File Directories 66
CMS Command Search Order 67
CMS Command Execution Characteristics 68
Displaying a List of Your CMS Files 70
Finding Files in Your FILELIST List 70
Erasing Files from FILELIST 72
Listing your Files with the LISTFILE command 72
Comparing Contents of Files 73
Copying Files 73
Renaming Files 73
Changing the Record Format of a File 74
Using Synonyms 75
Using Translations 76

Chapter 4. What You Can Do with CMS Commands 77
Beginning and Ending Your Terminal Session 78
Tailoring Your System 79
Requesting Information 81
Communicating with Other Computer Users 82
Controlling Terminal Output 83
Sharing Virtual Disks 84
What You Can Do to the Files in Your Virtual Reader 85
Receiving or Loading Files onto Your Disk 86
Erasing Files from Your Virtual Disk 87
Creating and Modifying Files 88
Moving Files 89

Chapter 5. Using the HELP Facility 91
Getting HELP on Commands 92
BRIEF HELP 93
DETAIL HELP 94
RELATED HELP 95
Getting HELP on SET and QUERY 97
Facts About Command HELP 98
Getting HELP on Messages 98
Menus 99

TASK Menus in HELP ... 99
Component Menus ... 100
Using the PA2 and PF Keys ... 101
Toggling ... 101
Using the MOREHELP Command ... 104
The System Product Editor ... 104
Printing HELP Screens ... 105
Working with Your HELP Files ... 106

Part 2: Working with VM/SP ... 107

Chapter 6. Editing Your Files ... 109
The System Product Editor ... 109
The XEDIT Command ... 110
Writing a File Onto Disk ... 111
What To Do When You Run Out of Space ... 114
Using the Editor in Line-Mode ... 117

Chapter 7. Using Real Printers, Punches, Readers, and Tapes ... 119
CMS Unit Record Device Support ... 119
Using the CP Spooling System ... 119
Spool File Characteristics ... 120
Altering Spool Files ... 122
Using Your Card Punch and Card Reader in CMS ... 124
Handling Tape Files in CMS ... 126
Using the CMS TAPE Command ... 127
Tape Labels in CMS ... 130
User Responsibilities ... 131
Label Processing in OS Simulation ... 131
Label Processing in CMS/DOS ... 140
CMS TAPESL Macro ... 143
Tape Label Processing by CMS Commands ... 143
LABELDEF Command ... 145
End-of-Volume and End-of-Tape Processing ... 146
Error Processing ... 149
The MOVEFILE Command ... 150
OS Utility Programs ... 150
Specifying Special Tape Handling Options ... 151
Moving Data Between Shared and Non-Shared Tape Subsystems ... 153

Chapter 8. Communicating with Other Computer Users ... 155
What is a Names File? ... 155
Creating a Names File ... 156
Sending Messages ... 160
Receiving Messages ... 161
Sending Notes and Files ... 163
Composing Notes ... 163
Sending a Note ... 164
Sending Files ... 165
Sending One File ... 167
Receiving Notes and Files ... 168

Chapter 9. Looking at VM/SP Through Windows ... 173

What Are Windows and Virtual Screens? 173
What's in a Window? 175
Using Full-Screen CMS 175
Status Information 177
Location Information 178
Your Default Windows and Virtual Screens 179
Special Keys 181
Messages in Full-Screen CMS 186
Re-Entering Commands 190
Working with Border Commands 192
Using the WM Window 198

Chapter 10. Using the CMS Batch Facility 205
Submitting Jobs to the CMS Batch Facility 205
Input to the Batch Machine 206
Submitting Virtual Card Input to the CMS Batch Facility 206
Other Input Records 209
How the Batch Facility Works 209
Preparing Jobs for Batch Execution 210
Restrictions on CP and CMS Commands in Batch Jobs 211
Batch Facility Output 212
Purging and Reordering Batch Jobs 213
Using EXEC Files for Input to the Batch Facility 214
Sample System Procedures for Batch Execution 215
A Batch EXEC for a Non-CMS User 217

Part 3: Learning to use EXECs 219

Chapter 11. Introduction to the EXEC Processors 221
The System Product Interpreter 221
The EXEC 2 Processor 222
The CMS EXEC Processor 223
Relationship of the EXEC interpreters 223
Invoking EXECs 224
Attributes of EXEC Files 224

Chapter 12. Creating System Product Interpreter EXECs 225
Creating a System Product Interpreter EXEC 225
Invoking Your EXEC Files 226
Sample System Product Interpreter EXECs 227

Chapter 13. Creating a PROFILE EXEC 229

Chapter 14. Commands Used with System Product Interpreter EXECs 233

Part 4: Tailoring Your System 245

Chapter 15. Customizing Full-Screen CMS 247
Tailoring System Defaults 248
POSITION WINDOW 251
SIZE WINDOW 252
MAXIMIZE WINDOW and RESTORE WINDOW 253

Using the SET Command 255
SET BORDER 255
SET RESERVED 257
SET WINDOW 259
Exploring Other Commands 261
Window and Virtual Screen Tables 262
Considerations When Disconnecting and Reconnecting 265
Message Routing Tables 266

Chapter 16. Tailoring the HELP Facility 267
Creating HELP Files 267
Filenames for HELP Files 267
Filetypes for HELP Files 268
Examples of Naming Conventions 270
Creating Menus for HELP Files 271
Creating Command HELP Files 275
Using Command Abbreviations 279
Adding Your Own HELP Components 279
Using HELPCONV to Create HELP Files 280
Changing Existing HELP Files 291
Adding HELP Files 291
Deleting HELP Files 292
Altering Existing HELP Files 292
Changing Menus 292

Appendixes 293

Appendix A. Considerations for Line Mode Terminals 295

Appendix B. Summary of CMS Commands 299

Appendix C. Summary of CP Commands 311

Appendix D. Considerations for Full-Screen CMS and Windowing 321

Summary of Changes 337

Glossary of Terms and Abbreviations 349

Bibliography 353

Index 359

The following information has been relocated:	
Programming for the CMS Environment Developing OS Programs Under CMS Developing VSE Programs Under CMS Using Access Method Services and VSAM Under CMS and CMS/DOS	Refer to VM/SP CMS for System Programming SC24-5289
Debugging Your Program Using VM/SP	Refer to VM Diagnosis Guide LY24-5241

Refer to the "Summary of Changes" for further details.

Figures

1. Sample of VM/SP Logo on a 3270-Type Terminal 7
2. Sample Status Notice in VM 15
3. Sample Status Notice in Full-Screen CMS 18
4. Sample XEDIT Screen 21
5. Sample XEDIT Screen In INPUT Mode 22
6. 3270 Screen Display 30
7. VM/SP Environments and Mode Switching 32
8. Filetypes Used by CMS Commands 51
9. Filetypes for Temporary Work Files 56
10. CMS Command Execution Characteristics 69
11. Sample FILELIST Screen 70
12. Sample of BRIEF HELP for the SENDFILE Command 94
13. Sample of DETAIL HELP for the SENDFILE Command 95
14. Sample of RELATED HELP for the ERASE Command 96
15. Sample HELP TASK Menu 100
16. Sample Component MENU for REXX 101
17. Toggling Between Layers of HELP 102
18. PA and PF Keys in the HELP Facility 103
19. Sample XEDIT Screen 112
20. Sample XEDIT Screen in INPUT Mode 113
21. Sample FILELIST Screen for a Particular Filetype 114
22. CP QUERY Unit Record Response 119
23. Sample NAMES Screen 156
24. Sample Entry for a List of Names 157
25. Another Sample Entry for a List of Names 158
26. Sample Entry for a Chained List of Names 159
27. Sample ‘userid NAMES’ File 160
28. Sample Note with Short Headings 164
29. Sample SENDFILE Menu 166
30. Sample FILELIST Screen Invoked from SENDFILE 167
31. Sample RDRLIST Screen 169
32. Sample RDRLIST Screen after Receiving a File 170
33. Sample RDRLIST Screen after Receiving a Note 171
34. A Window into a Virtual Screen 174
35. Full-Screen CMS 176
36. Default Windows and Virtual Screens 179
37. CMSPF Key Settings 182
38. Displaying the CMSPF Key Settings 183
39. Setting CMSPF 9 to TELL 184
40. WMPF Key Settings 186
41. Adding an Entry to the Names File 188
42. The MESSAGE Window in the Names File 189
43. Popping the MESSAGE Window 191
44. Looking at the Corners of a Window Border 193
45. Using a Border Command to Scroll Forward 194
46. The Result of Scrolling Forward 195

47. Scrolling Backward through a Window Border 196
48. Scrolling to the Right through a Window Border 197
49. Scrolling to the Left through a Window Border 198
50. Popping the WM Window 199
51. Dropping the MESSAGE Window 200
52. Displaying the MESSAGE Window 201
53. Changing the MESSAGE Window 202
54. The WM Window 203
55. Restoring the MESSAGE Window 204
56. Your Newly-Defined MESSAGE Window 250
57. Using the POPDROP EXEC 251
58. Using the POSITION WINDOW Command 252
59. "Popping" the MESSAGE Window 253
60. "Maximizing" a Window 254
61. The Window after RESTORE WINDOW 255
62. Changing Only the Top Border 256
63. Changing the Bottom and Left Window Borders 257
64. Deleting a Window Title 258
65. Deleting a Blank Reserved Line 259
66. Message Class Indicator 260
67. The MESSAGE Window 261
68. Default Windows 262
69. Default Virtual Screens 264
70. Default Settings for Message Routing 266
71. Sample of HELPMENU File Creation 273
72. Sample of HELPTASK File Creation 275
73. Format of the RELATED Section of the ERASE Command 278
74. HELP and HELPCONV Format Word Summary 282
75. CMS Command Summary 302
76. CMS Commands for System Programmers 308
77. CP Privilege Classes 312
78. CP Command Summary 313
79. CMS Command Summary 322
80. Default Virtual Screens 324
81. Default Windows 324
82. Screen Organization 326
83. CMSPF Key Settings 327
84. Default Settings for Message Routing 328
85. WMPF Key Settings 329
86. Restructuring of the CMS User's Guide 337

Part 1: Getting Acquainted with VM/SP

Learning how to use CMS is not an end in itself: you have specific tasks to do, and you need to use the computer to perform them. VM/SP has been designed to make these tasks easier, but if you are unfamiliar with VM/SP, then the tasks may seem more difficult. The information contained in Part 1 of the *VM/SP CMS User's Guide* is organized to help you make the acquaintance of VM/SP quickly, so that it enhances, rather than impedes, the performance of your tasks.

Chapter 1: Introduction to VM/SP introduces you to VM/SP and its conversational component, CMS. It should help you to get a picture of how you, at a terminal, use and interact with the system.

Chapter 2: VM/SP Environments and Mode Switching. During a terminal session, commands and requests that you enter are processed by different parts of the system. How and when you can communicate with these different programs is described in this chapter.

Chapter 3: The CMS File System. Almost every CMS command that you enter results in some kind of activity with a direct access storage device (DASD), known in CMS simply as a disk, or minidisk. Data and programs are stored on disks in what are called "files". This chapter introduces you to the creation and handling of CMS files.

Chapter 4: What You Can Do with CMS Commands contains a sampling of commands in various functional areas to give you a general idea of the kinds of things you can do and the commands available to help you do them.

Chapter 5: Using the HELP Facility. The CMS HELP facility provides an online display of documentation and an online list of tasks that guide you to the appropriate HELP file for the command or commands. This chapter describes the HELP facility and shows you how to use HELP to assist you in performing CMS tasks.

Chapter 1. Introduction to VM/SP

Virtual Machine/System Product (VM/SP) is a program product that controls "virtual machines". A virtual machine is the functional equivalent of a real computer that you control from your terminal, using a command language of verbs and nouns.

The command languages correspond to the components of VM/SP. CP controls the resources of the real machine; that is, the physical machine in your computer room; it also manages the communications among virtual machines and between a virtual machine and the real system. CMS is the conversational operating system designed specifically to run under CP; it can simulate many of the functions of OS (Operating System) and DOS (Disk Operating System), so that you can run many OS and DOS programs in a conversational environment.

Although this publication is concerned primarily with using CMS, it also contains examples of CP commands with which you, as a CMS user, should be familiar.

How You Communicate With VM/SP

When you are running your virtual machine under VM/SP, each command, or request for work, that you enter on your terminal is processed as it is entered; usually, you enter one command at a time and commands are processed in the order that you enter them.

You can enter CP commands from either the CP or CMS environment; but you cannot enter CMS commands while in the CP environment. The concept of "environments" in VM/SP is discussed in Chapter 2, "VM/SP Environments and Mode Switching."

After you have typed or keyed in the line you wish to enter, you press the Return or ENTER key on the keyboard. When you press this key, the line you have entered is passed to the command environment you want to have process it. If you press this key without entering any data, you have entered a "null line". Null lines sometimes have special meanings in VM/SP.

If you make a mistake entering a command, VM/SP tells you what your mistake was, and you must enter the line again. The examples in this publication assume that the commands are correctly entered.

You can enter commands using any combination of uppercase and lowercase characters; VM/SP translates your input to uppercase. Examples in this publication show all user-entered input lines in blue type; system responses are shown in black.

The CP Command Language

You use CP commands to communicate with the control program. CP commands control the devices attached to your virtual machine and their characteristics.

For example, if you want to allocate additional disk space for a work area or if you want to increase the virtual address space assigned to your virtual machine, use the CP command DEFINE. CP takes care of the space allocation for you and then allows your virtual machine to use it.

Or if, for example, you are receiving printed output at your terminal and do not want to be interrupted by messages from other VM/SP users, you can use the CP command SET MSG OFF to refuse messages, because it is CP that handles communication among virtual machines. The CP QUERY SET command displays the status of the CP SET MSG function and other CP SET command functions.

Using CP commands, you can send messages to the system operator and to other users, or you can modify the configuration of devices in your virtual machine. CP commands are available to all virtual machines using VM/SP. You can invoke these commands when you are in the virtual machine environment using CMS (or some other operating system) in your virtual machine.

The CP commands and command privilege classes (not all commands are available to all users) are listed in Appendix C, "Summary of CP Commands." The CP Commands applicable to the average user are discussed in detail in the *VM/SP CP Command Reference.* However, because many CP commands are used with CMS commands, some of the CP commands you will use most frequently are discussed in this publication, in the context of their usefulness for a CMS application.

The CMS Command Language

The CMS command language allows you to create, modify, and debug problem or application programs and, in general, to manipulate data files.

Many OS language processors can be executed under CMS: the assembler, VS BASIC, OS FORTRAN, VS FORTRAN, OS/VS COBOL, and OS PL/I Optimizing and Checkout Compilers. In addition, the DOS/VS COBOL, DOS PL/I, and VS APL Program Products are supported. You can find a comprehensive list of language processors that can be executed under CMS and relevant publications in the *VM/SP Introduction.* CMS executes the assembler and the compilers when you invoke them with CMS commands.

When you issue the XEDIT command, you invoke the System Product Editor to create, modify, or manipulate CMS disk files. Once the VM/SP System Product Editor has been invoked, you may execute XEDIT subcommands and use the System Product Interpreter or EXEC 2 macro facility. The System Product Interpreter, CMS EXEC interpreter, and the EXEC 2 interpreter provide execution procedures consisting of CP and CMS commands; they also provide the conditional execution capability of a macro language.

Other CMS commands allow you to read cards from a virtual card reader, punch cards to a virtual card punch, and print records on a virtual printer. Many commands are provided to help you manipulate your virtual disks and files.

In addition, you can use windowing commands and full-screen CMS to help you manage the data on your physical screen. When you set full-screen on, you can type commands from almost anywhere on the physical screen. Full-screen CMS also allows you to scroll forward and backward through your CMS session to see commands you entered previously and CMS responses to those commands.

You use the HELP command to display at your terminal information on how to use CP commands and CMS commands, subcommands, and EXECs, and explanations of CP and CMS messages. You can issue the HELP command when a brief explanation of syntax, a parameter, or function is sufficient, thereby avoiding interrupting your terminal session to refer to a manual.

You can also invoke CP commands from within the CMS virtual machine environment.

If your VM/SP system supports a language other than English, you can receive messages, view productivity aid panels (like the FILELIST screen), and enter various CMS commands in that language.

You can use the QUERY LANGLIST command to find out the languages that your virtual machine supports. You can also find out what language environment you are currently working in with two QUERY commands:

1. The QUERY CPLANG command tells you the language environment for CP.

2. The QUERY LANGUAGE command tells you the language environment for CMS.

VM/SP allows you to change the language you are working in without having to quit your session. The SET LANGUAGE command automatically gets all the information you need to interact with VM/SP in another language. SET LANGUAGE also allows you to add language information for applications.

Refer to the *VM/SP CMS Command Reference* for more detailed information about the SET LANGUAGE, QUERY LANGLIST, and QUERY LANGUAGE commands. Refer to the *VM/SP CP Command Reference* for information on the QUERY CPLANG command.

If you want to know more about the languages available on your VM/SP system, you can contact your system administrator.

What You Must Know to Use VM/SP

Before you can use CP and CMS, you should know:

1. How to operate your terminal
2. Your userid (user identification) and password.

The Terminal: Your Virtual Console

There are many types of terminals you can use as a VM/SP virtual console. Before you can conveniently use any of the commands and facilities described in this publication, you have to familiarize yourself with the terminal you are using. Generally, you can find information about the type of terminal you are using and how to use it with VM/SP in the *VM/SP Terminal Reference.* If your terminal is a 3767, you also need the *IBM 3767 Communication Terminal Operator's Guide, GA18-2000.*

In this publication, examples and usage notes assume that you are using a display terminal (such as a 3277). If you are using a typewriter style terminal (such as a 2741) consult Appendix A, "Considerations for Line Mode Terminals" for a discussion of special techniques that you can use to communicate with VM/SP.

Your Userid and Password: Keys into the System

Your userid is a symbol that identifies your virtual machine to VM/SP and allows you to gain access to the system. Your password is a symbol that functions as a protective device ensuring that only those allowed can use your virtual machine. The userid and password are usually defined by the system programmer for your installation.

Beginning Your Terminal Session

To establish contact with VM/SP, you switch the terminal device on and VM/SP responds with a logo and some form of the message:

```
VIRTUAL MACHINE/SYSTEM PRODUCT
```

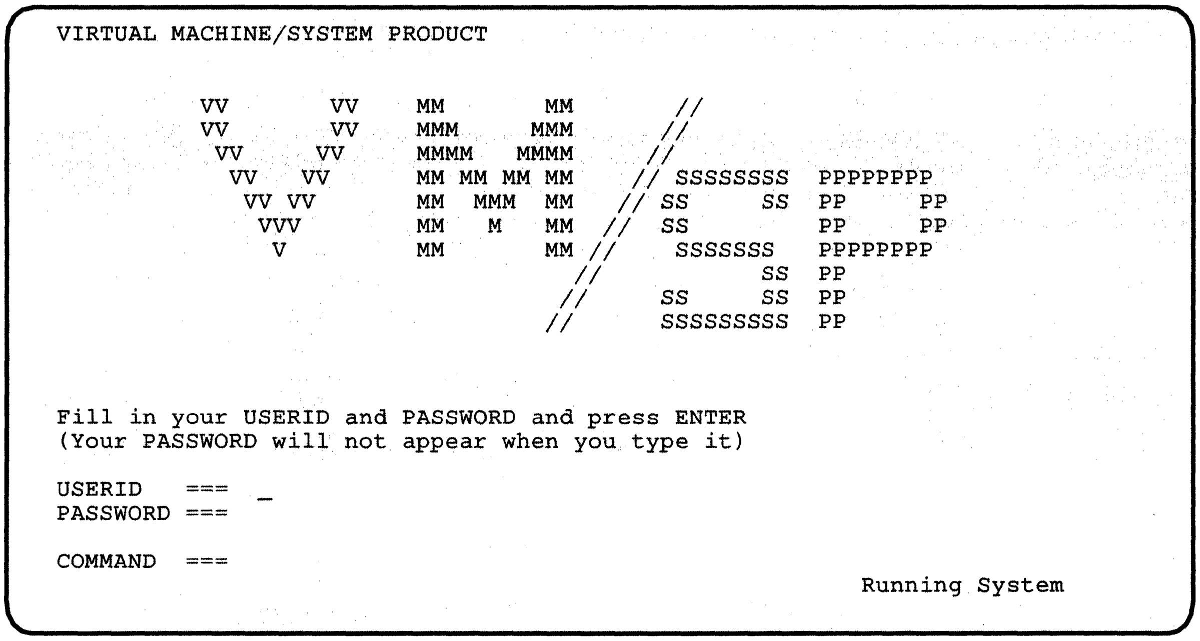
to let you know that VM/SP is running and that you can use it. If you do not receive the "VIRTUAL MACHINE/SYSTEM PRODUCT" message, see the *VM/SP Terminal Reference* for specific directions.

Note: If your terminal is not a 3270-type, use the logon procedures described in the section entitled "Logon Exceptions" on page 8.

Logging On at a 3270-Type Terminal

If you are using a 3270-type terminal, you may log on directly from the logo screen.

In the following figure, you will notice that below the actual VM/SP logo on the logo screen are two lines instructing you to fill in your userid and password. Following these instructions are three input lines labeled USERID, PASSWORD, and COMMAND. The cursor is placed at the input line for USERID.



```
VIRTUAL MACHINE/SYSTEM PRODUCT

          VV        VV   MM         MM        //
          VV        VV   MMM       MMM       //
           VV      VV    MMMM     MMMM      //
            VV    VV     MM MM MM MM      // SSSSSSSS  PPPPPPPP
             VV VV       MM  MMM  MM     // SS      SS  PP      PP
              VVV        MM   M   MM    //  SS          PP      PP
               V         MM       MM   //    SSSSSSS    PPPPPPPP
                                      //           SS   PP
                                     //   SS      SS    PP
                                    //    SSSSSSSSS     PP


Fill in your USERID and PASSWORD and press ENTER
(Your PASSWORD will not appear when you type it)

USERID   === _
PASSWORD ===

COMMAND  ===
                                                       Running System
```

Figure 1. Sample of VM/SP Logo on a 3270-Type Terminal

You may type your userid and password in the USERID and PASSWORD input areas and press ENTER. If all of the information is entered correctly, the logo is cleared from the screen, no further prompts will appear, and you will be logged on to the system. If an invalid userid or password is entered, the logo is cleared from the screen, and the following message and prompt will appear:

```
DMKLOG050E LOGON unsuccessful--incorrect password
```

or

```
DMKLOG053E userid not in CP directory
```

```
Enter one of the following commands:

   LOGON userid           (Example:  LOGON VMUSER1)
   DIAL userid            (Example:  DIAL VMUSER2)
   MSG userid message     (Example:  MSG VMUSER2 GOOD MORNING)
   LOGOFF
```

If you enter only your PASSWORD in the input area, or if your USERID, as entered, contains one or more blanks (for example, V MUSER1), the following error message will be issued, followed by the LOGON prompts:

```
DMKCFM288E  LOGON from the initial screen was unsuccessful
```

You may also enter your userid in the USERID input area, without your password or enter the LOGON command, followed by your userid, in the COMMAND input area. The following prompt will appear:

```
Enter password (it will not appear when typed):
```

If you have entered the information correctly, the logo is cleared from the screen and you will be logged on to the system.

Logon Exceptions

If your terminal is not a 3270-type, and the VM/SP logo screen is displayed, you can now press the ENTER key (or equivalent) on your terminal to clear the display. Now, enter your first command to identify yourself to VM/SP, the CP LOGON command. If your userid is TIGER, then you type:

```
logon tiger
```

and press the ENTER key. You only need to type L, because L is short for LOGON.

If VM/SP accepts your userid, it responds by asking you for your password:

```
Enter password (it will not appear when typed):
```

Now, carefully type your password, and press the ENTER key. You may not see your password as you type it. This is a security measure; it prevents others from learning your password. If you receive the message:

```
DMKLOG050E LOGON unsuccessful--password incorrect
```

followed by:

```
Enter one of the following commands:

   LOGON userid          (Example:  LOGON VMUSER1)
   DIAL userid           (Example:  DIAL VMUSER2)
   MSG userid message    (Example:  MSG VMUSER2 GOOD MORNING)
   LOGOFF
```

you will have to start over, beginning with the CP LOGON command. You will also receive a prompting message to help you restart if you make a mistake in entering other logon information.

Getting Into CMS

After a successful logon, your next step is to load CMS in your virtual machine using the CP IPL command. IPL stands for Initial Program Load,

```
ipl cms
```

where "cms" is assumed to be the saved system name for your installation's CMS. VM/SP responds by displaying a message such as:

```
VM/SP CMS - 05/16/86  12:54
```

to indicate that the IPL command executed successfully. Press the ENTER key again. VM responds with a message. The last line, known as the ready message, may look like this:

```
Ready; T=0.01/0.01 08:05:50
```

At this point you have loaded CMS, and you can now enter both CP and CMS commands.

Your userid may be set up for an automatic IPL, so that you receive a message, indicating that you are in the CMS command environment, without having to issue the IPL command.

Note: If this is the first time you are using a new virtual disk assigned to you, and you receive the message:

```
DMSACC112S A(191) device error
```

you must "format" the disk, that is, prepare it for use with CMS files. See the section entitled "Formatting Virtual Disks" on page 24.

Ending Your Terminal Session

To end your terminal session, use the CP LOGOFF command. Enter:

```
logoff
```

and press the ENTER key, or just enter:

```
log
```

and press the ENTER key, because LOG is short for LOGOFF.

At times you may be running a long program under one userid and wish to use your terminal for some other work. Then, you can disconnect your terminal using the CP command DISCONN:

```
#cp disconn

  or

#cp disconn hold
```

Your virtual machine continues to run while disconnected. Disconnected virtual machines are automatically logged off after 15 minutes if a CP READ or VM READ status is pending, that is, if the machine is waiting for you to enter a response. If you want your virtual machine to be logged off the system 15 minutes after your program has finished executing, issue SET AUTOREAD ON. AUTOREAD ON is the default for some terminal types; see the *VM/SP CMS Command Reference* for a description of the SET AUTOREAD command. Refer to the *VM/SP Terminal Reference* information on specific terminal types and how to reconnect to your virtual machine.

If you want to regain terminal control of your virtual machine after disconnecting, log on as you would to begin your terminal session. Your virtual machine is placed in the CP environment, and to resume its execution, you use the CP command BEGIN. You should not disconnect your virtual machine if a program requires an operator response, because the console read request cannot be satisfied.

Entering Commands

The IBM 3270 display terminal, commonly referred to as a 3270, functions somewhat differently from a typewriter-style terminal when you use it as a virtual machine console under VM/SP. Apart from the obvious difference in the way output is displayed, there are special techniques you can use with a 3270 that you cannot use on a 2741 or other typewriter terminals. Since the keyboard on a 3270 is never locked during the execution of a command or program, you can enter successive commands without waiting for the completion of the previous command. This stacking function can be combined with the other methods of stacking lines, such as using the logical line end symbol (#) to stack several commands. If you try to enter more lines than the terminal buffer can accommodate, however, you receive the status message NOT ACCEPTED and you must wait until the buffer is cleared before you can enter the line.

You will find, as you become accustomed to using a 3270, that the #CP function is very useful. The #CP function allows you to pass a command to the control program immediately, bypassing any processing by the virtual machine (CMS). The #CP function can be used in any VM/SP environment, and you can enter it even when a program is executing. You do not have to interrupt a program's execution to enter a command such as:

```
#cp query reader all
```

to display the contents of your reader, or:

```
#cp spool printer class s
```

to spool your virtual printer.

RETRIEVE Function

One of the most common user difficulties is typing errors. The RETRIEVE function provides a convenient and time-saving method of correcting errors without retyping the entire input. You can use this function by defining a program function (PF) key for it, using a command such as:

```
set pf6 retrieve
```

If you define a PF key for the RETRIEVE function, VM/SP remembers each input line entered at the terminal. When you press the PF key, VM/SP redisplays the latest input line in the input area, so that you can modify and re-enter the data. This allows you to correct errors, change your input, or repeatedly reissue a command.

VM/SP actually remembers several input lines. The number of lines remembered depends on the length of the lines; VM/SP remembers more short lines than long lines, but it can always remember at least one full input line. Duplicate input lines (lines that are the same as the previous input) are not remembered because it is not useful to remember the same line twice. For security reasons, input lines that are not displayed at the terminal, such as passwords, are never remembered.

When a RETRIEVE program function key is first pressed, VM/SP redisplays the latest input line. If a RETRIEVE key is pressed again, VM/SP displays the previous input line. As the key is pressed, VM/SP steps through the input lines displaying them one at a time. When VM/SP reaches the oldest line that it has remembered, it cycles back to the latest one again. When an input line is entered, VM/SP resets itself so that the RETRIEVE program function key starts with the latest input line.

If you are using full-screen CMS, you will not need to define a PF key for the RETRIEVE function since the default setting for CMSPF 6 is RETRIEVE. Full-screen CMS also provides you with another method to retrieve commands which you previously entered. Simply scroll your screen back to a command you wish to re-enter, position the cursor over the command, re-type any character, and press ENTER. The command will be re-issued.

Note: If you press the RETRIEVE key too many times, you may go past the command you wanted to re-enter. For CP and CMS commands, there is a simple way to reset the RETRIEVE function to the latest input line: simply enter a single asterisk (*), which is treated as a comment by both CP and CMS. Then press the RETRIEVE program function key once to get the asterisk redisplayed, and a second time to get the previous line redisplayed.

Setting Program Function Keys

If there are commands that you use frequently, you can set the program function (PF) keys on your terminal to execute them. Although there is one set of function keys (1 through 24) on your terminal, these keys can have different settings in various environments.

For example, when you first logon, you might set your PF keys to perform certain functions. Then, when you enter different CMS environments, your PF keys may have entirely different settings. Chapter 9, "Looking at VM/SP Through Windows," provides details on PF keys in full-screen CMS and in the WM environment. The remainder of this section will concentrate on setting PF keys for use when full-screen CMS is set off.

Some examples of commands you might wish to catalog on your CP and CMS PF keys are:

```
#CP QUERY READER ALL
#CP QUERY PRINTER ALL
QUERY SEARCH
```

To set function keys 1, 2, and 3 to perform these command functions, enter:

```
cp set pf1 immed "#cp query reader all
cp set pf2 immed "#cp query printer all
cp set pf3 immed query search
```

Note: When you want to execute a #CP function with a PF key, or you want a PF key to execute a series of commands, you must use the logical escape symbol (") when you enter the SET command.

You can change a PF key setting any time during a terminal session, according to your needs. This example shows how you can change the setting of the PF5 key:

```
cp set pf5 immed xedit test file"#bo"#input line"#file
```

sets the PF5 key as:

```
XEDIT TEST FILE#BO#INPUT LINE#FILE
```

Then, when you press PF5, VM/SP will XEDIT the file TEST FILE, input the word "line", and write the file to your A-disk.

You can also set all of your program function keys in your PROFILE EXEC so they are set each time you load CMS. To change the setting of the PF5 key in your PROFILE EXEC, you could add to your PROFILE EXEC the line:

```
CP SET PF5 IMMED XEDIT TEST FILE #BO# INPUT LINE #FILE
```

Then, the next time you load CMS, the PF5 key will be set to perform this function.

Note: In this instance, you would not need to include the logical escape characters because the command was entered from a file.

The above examples use the IMMED operand of the SET command, which specifies that the function is performed as soon as you press the PF key. You can also set a key so that it is delayed; that is, the command or data line is placed in the user input area. Then, you must press the ENTER key to execute the command. You may modify the line before you enter it. This is the default setting (DELAY) for program function keys. For example, you might set a key as:

```
QUERY DISK X@
```

When you press this PF key, the command is placed in the user input area, with the cursor positioned following the "@" logical character delete symbol; you can enter the mode letter of the disk you are querying before you press the ENTER key to execute the command. If you enter "A", the "X" is deleted, and the resulting command as seen by CMS is QUERY DISK A. For more information on using the logical character delete symbol, refer to Appendix A.

For more details on setting PF keys, see the *VM/SP CP Command Reference* and the *VM/SP Terminal Reference.*

Using Full-Screen CMS

Another way to facilitate your work is to use full-screen CMS. You can do this by entering on the command line, SET FULLSCREEN ON or by putting this command in your PROFILE EXEC. Your system administrator may also put this command in your system profile (SYSPROF) EXEC. In this case, you will automatically be in full-screen CMS when you IPL CMS. For more information on SYSPROF, refer to *VM/SP CMS for System Programming.*

You can take advantage of full-screen CMS capabilities to perform several functions such as entering commands from almost anywhere on your physical screen and displaying messages and other information in windows on your screen.

For more information on how to use full-screen CMS and windowing support, see Chapter 9, "Looking at VM/SP through Windows."

Display Screen Characteristics

Messages

During a CP or CMS session (other than an edit session) messages and warnings from the system operator or other users are highlighted. This distinguishes these messages from other output and lessens the possibility of important messages being lost or ignored.

A major feature of a 3270 display screen is the screen status area, which indicates, at all times that you are logged on, the current operating condition your virtual machine is in. Understanding the status conditions can help you use CMS on a 3270 more effectively.

VM Status Notices

The screen status area indicates one of the following conditions:

CP READ
This status notice is the first one you see; it indicates that the terminal is waiting for a line to be read by the control program. You can enter only CP commands when the screen status area indicates a CP READ.

VM READ
This status notice indicates that your terminal is waiting for a line to be issued to your virtual machine; you may be in the CMS environment, in the edit or debug environments, or you may be executing a program or an EXEC that has issued a read to the console. Following is an example of this status notice:

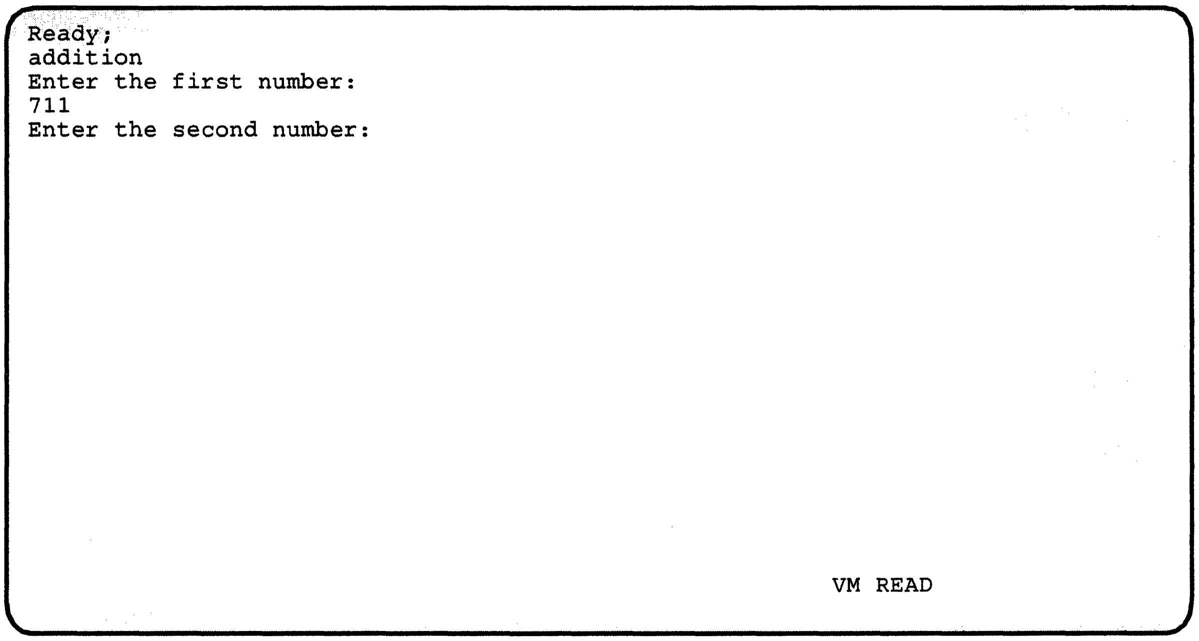

```
Ready;
addition
Enter the first number:
711
Enter the second number:















                                                       VM READ
```

Figure 2. Sample Status Notice in VM

While in VM READ, you can pass null lines or Immediate commands to CMS. This procedure is particularly useful when you wish to stop execution of an EXEC that is issuing VM READs to the terminal. You can type HI or HX and have it passed to CMS without being read and interpreted as data by the EXEC. At the VM READ, use the cursor movement key to move the cursor back one space from its current position at the command line. (This results in the cursor being positioned in the lower right corner of the screen, three lines up from the bottom). Press the ENTER key. The cursor will return to the command line, and your terminal will remain in VM READ status. At this point, you can enter HI, HX, or any Immediate command, or if you wish to resume execution of the EXEC, enter the next line of data.

Refer to the *VM/SP CMS Command Reference* for more information on using HI or HX.

RUNNING

This status notice indicates that your virtual machine is operating. Once you have loaded CMS and are using the CMS environment, this status is almost continually in effect, even when you are not currently executing a command or program.

You can alter the way this works by using the AUTOREAD function of the SET command. When the AUTOREAD setting is OFF, (the default for display terminals), your terminal displays a RUNNING status after the execution of each CMS command. If you want the terminal to be in a VM READ status following each command, issue:

```
set autoread on
```

The ON setting is the default for typewriter terminals, because a read on a typewriter terminal must be accompanied by the unlocking of the keyboard.

The advantage of keeping your virtual machine in a running status even when it is not actually executing a program is that it makes your terminal ready to receive messages. If your terminal is waiting for a read, either from CP or from the virtual machine, and if a user or a program sends a message to your virtual console, then the message is not displayed until you use the ENTER key to enter a command or null line. When your machine is in a running status, the terminal console is always ready to accept messages.

If your virtual machine is in the CP environment, and you want your terminal to be in a running status, you can use the command:

```
sleep
```

To return to the CP READ status, you must press the PA1 key or the ENTER key.

MORE...

This status notice indicates that your display screen is full, but that there is more data to be displayed. This message, in addition to indicating that there is more data, gives you a chance to freeze your screen's current display so you can continue to examine it, if necessary.

When you see the screen is in a MORE... status, you can either:

- Press the Clear, Cancel, or PA2 keys to clear the screen and see the next screen, or
- Press the ENTER key to put the screen in HOLDING status.

If you do not do either, then after 60 seconds, the screen is cleared and the next screen is displayed.

HOLDING

This status notice indicates that you have pressed the ENTER key to freeze the screen. You must use the Cancel, Clear, or PA2 keys to erase this screen and go on to the next display.

A holding status also results if you have received a message that appeared on this screen. When the screen becomes full, it does not automatically pass to the next display after 60 seconds, but waits until you specifically clear the screen. (This feature ensures that any important messages you receive are not lost.)

NOT ACCEPTED
This status notice indicates that you are trying to enter a command line but the terminal buffer is full and cannot accept it. This message is also issued when you attempt to use the 3270 COPY function and a printer is either not available or not ready.

Full-Screen CMS Status Notices

Once you enter the command SET FULLSCREEN ON, the following status notices are displayed:

Executing a command
This status notice indicates that the system is processing your command.

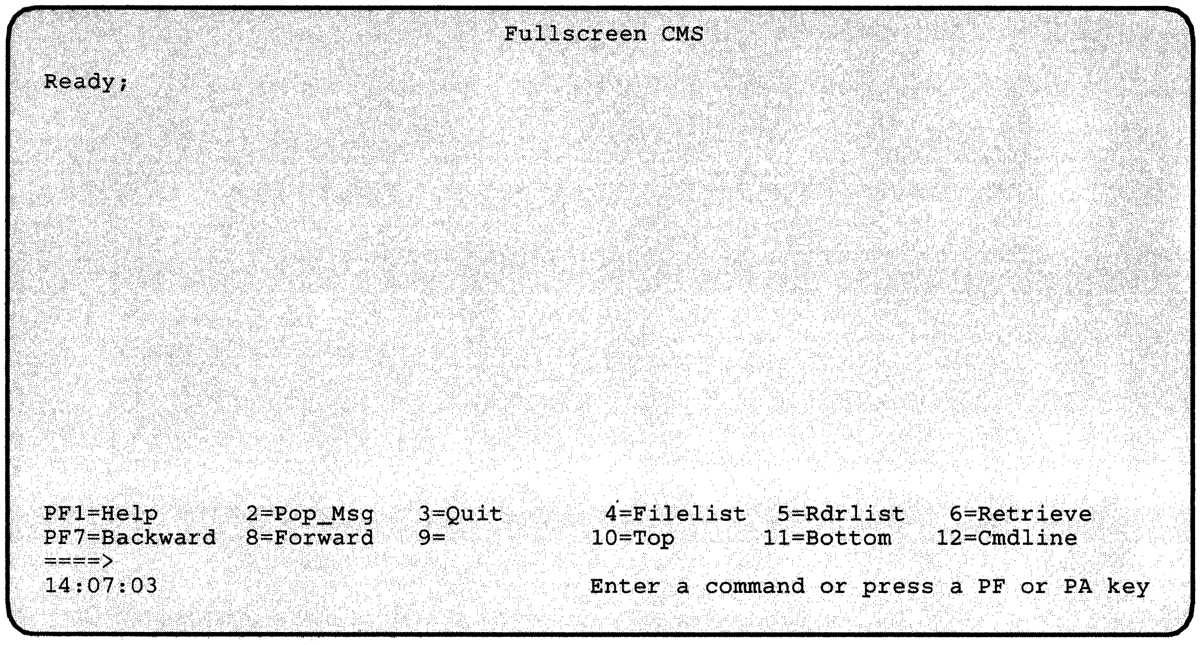
Enter your response in vscreen *'vname'*
This status notice indicates that the system is waiting for your reply to a request.

Note: In this message, *'vname'* will be replaced by the name of the virtual screen in which you are to enter your response.

Scroll for more information in vscreen *'vname'*
This status notice indicates that the system is waiting for you to scroll forward a window that is connected to the specified virtual screen.

Note: In this message, *'vname'* will be replaced by the name of a virtual screen.

Enter a command or press a PF or PA key
This status notice indicates that the system is waiting to process your next input. See the following example:



```
                              Fullscreen CMS
Ready;











PF1=Help       2=Pop_Msg    3=Quit        4=Filelist  5=Rdrlist    6=Retrieve
PF7=Backward   8=Forward    9=           10=Top       11=Bottom    12=Cmdline
====>
14:07:03                                 Enter a command or press a PF or PA key
```

Figure 3. Sample Status Notice in Full-Screen CMS

> *Note:* If you set autoread on, you will see the status notice "Enter your response in vscreen CMS" following each command.

Whenever there is a conflict between status notices, the highest priority status notice will be displayed. In the previous list, the status notices are listed in order of priority, from highest to lowest.

For more information on full-screen CMS, refer to Chapter 9, "Looking at VM/SP Through Windows."

Additional Display Screen Capabilities

Both extended highlighting and the seven-color feature are available on the 3279 Models 2 and 3. If you are using 3278 Model 2, 3, 4, or 5, the options for both features will be accepted. However, only the highlight feature is available.

The CP SCREEN command (with its operands) allows you to chose one of three highlighting features (blinking, underscore, or reverse video) and one of seven different colors (red, green, blue, pink, turquoise, yellow, or white) for each screen area.

If you want the input area to be turquoise without highlighting, you should enter:

```
screen inarea turquois none
```

Or, if you want the input area pink and the status area yellow with the blinking highlight, you should enter:

```
screen inarea pink status yellow blink
```

The CP QUERY SCREEN command displays the color and extended highlight values currently in effect. For more details on the CP SCREEN command, see the *VM/SP CP Command Reference* and for more details on the terminal display areas, see the *VM/SP Terminal Reference.*

You can tailor your XEDIT screen colors with the XEDIT subcommand SET COLOR. Refer to the *VM/SP System Product Editor Command and Macro Reference* for a description of the SET COLOR XEDIT subcommand.

When you are using CMS windowing support, you can tailor your screen for color, extended highlighting, Programmable Symbol Sets (PSS), character attributes, and Double-Byte Character Sets (DBCS). For more information, refer to the *VM/SP System Product Editor Command and Macro Reference* and the *VM/SP CMS Command Reference.*

How VM/SP Responds to Your Commands

CP and CMS respond differently to different types of requests. All CMS command responses (and all responses to CP commands that are entered from the CMS environment) are followed by the CMS ready message. The form of the ready message can vary, because it can be changed using the SET command. The long form of the ready message is:

```
Ready; T=7.36/19.89 09:26:11
```

If you have issued the command:

```
set rdymsg smsg
```

meaning, set the ready message to the short form, the ready message looks like:

```
Ready;
```

When you enter a command incorrectly, you receive a message describing the error. The ready message contains the last 5 digits (4 digits for a negative return code) from the command. For example:

```
Ready(00028);
```

indicates that the return code from the command was 28.

A ready message from the command may contain a negative return code; for example:

```
Ready(-0001);
```

indicates that the return code from the command was -1.

Some Sample CP and CMS Command Responses

If you enter a CP or CMS command that requests information about your virtual machine, the response should be the information requested. For example, if you issue the command:

```
query reader all
```

CP responds by showing you the contents of your reader, for example:

```
ORIGINID FILE CLASS RECORDS  CPY HOLD DATE  TIME     NAME    TYPE   DIST
BROWNL   1725 A PUN 00000009 001 NONE 05/22 10:59:10 BROWNL  NOTE  G67/33
NETTLE   0711 A PUN 00000016 001 NONE 05/21 13:02:54 NETTLE  NOTE  G42/02
```

Similarly, if you issue the CMS command:

```
listfile * assemble c
```

you might receive the following information:

```
JUNK     ASSEMBLE C1
MYPROG   ASSEMBLE C1
```

If you enter a CP command to alter your virtual machine configuration or the status of your spool files, CP responds by telling you that the task is accomplished. The response to:

```
purge reader all
```

might be:

```
0004 FILES PURGED
```

Some CP commands, those that alter some of the characteristics of your virtual machine, give you no response at all. If you enter:

```
spool e class x hold
```

you receive no response from CP.

Certain CMS commands may issue prompting messages, to request you to enter more information. The SORT command, which sorts CMS disk files, is an example. If you enter:

```
sort in file a1 out file a1
```

you are prompted with the message:

```
DMSSRT604R Enter sort fields:
```

and you can then specify which fields you wish the input records to be sorted on.

Working with CMS

If you have just logged on for the first time, and you want to try a few CMS commands, enter:

```
query disk a
```

the response might look like:

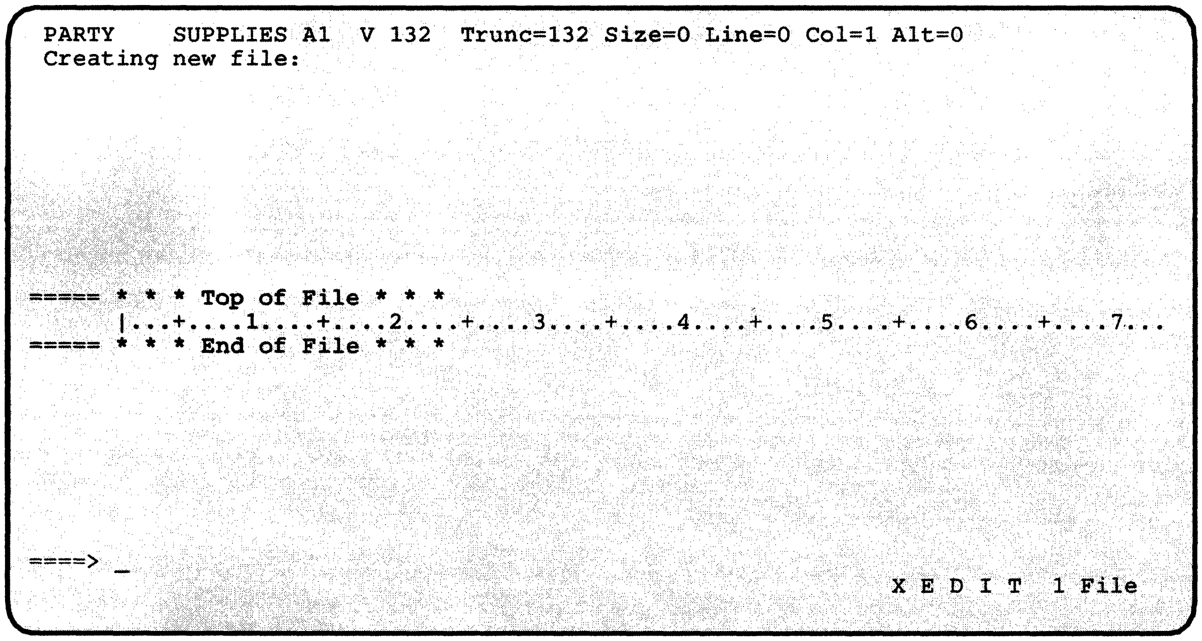
```
LABEL  CUU M STAT CYL TYPE BLKSIZE    BLKS USED-(%)  BLKS LEFT  BLKS TOTAL
PLC191 191 A  R/W  13 3380  1024           4864- 80       1181        6045
```

The response should tell you that you have an A-disk at virtual address 191; it also provides information such as how much room there is on the disk and how much of it is used. Again, if you receive an error message that indicates the disk may not be formatted, see the section entitled "Formatting Virtual Disks" on page 24.

Your A-disk is the disk you use most often in CMS, to contain your CMS files. Files are collections of data, and may have many purposes. You can invoke the System Product Editor to create and modify files with the XEDIT command. To create a file named PARTY SUPPLIES, enter:

```
xedit party supplies
```

The display will look like Figure 4.



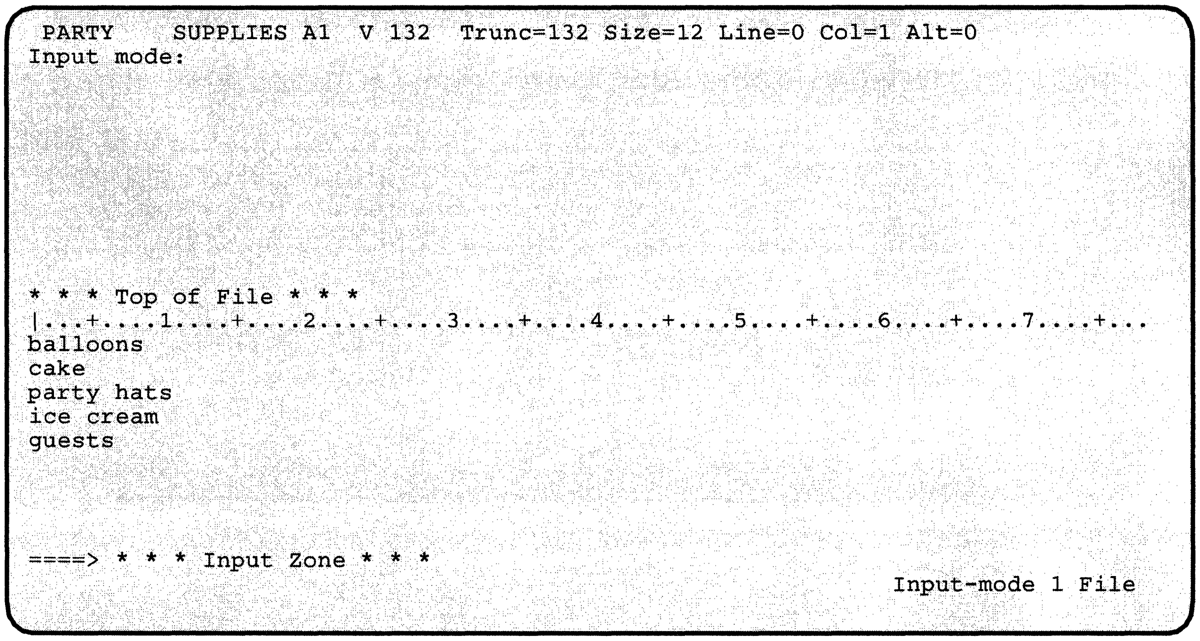
```
 PARTY    SUPPLIES A1  V 132  Trunc=132 Size=0 Line=0 Col=1 Alt=0
 Creating new file:










===== * * * Top of File * * *
      |...+....1....+....2....+....3....+....4....+....5....+....6....+....7...
===== * * * End of File * * *










====> _
                                                                 X E D I T  1 File
```

Figure 4. Sample XEDIT Screen

On the command line (next to the arrow) type INPUT and press the ENTER key. The file is placed in input mode. The cursor is placed automatically on the first line in the input zone, where you can enter your data. You are writing input lines that are eventually going to be written onto your A-disk.

Enter the following data:

```
balloons
cake
party hats
ice cream
guests
```



```
 PARTY    SUPPLIES A1  V 132  Trunc=132 Size=12 Line=0 Col=1 Alt=0
Input mode:









* * * Top of File * * *
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+...
balloons
cake
party hats
ice cream
guests



====> * * * Input Zone * * *
                                                                  Input-mode 1 File
```

Figure 5. Sample XEDIT Screen In INPUT Mode

Now, press the ENTER key, the screen moves up so that you can enter more data.

When you are finished entering data, press the ENTER key again to return to edit mode.

To keep this file in permanent storage, you type FILE on the command line and press the ENTER key. You should see a message that looks something like this:

```
Ready;
```

Even though the file has disappeared from your screen, the editor has saved it on your disk.

Let's check and see if the file was really saved. We'll use the LISTFILE command to list the files on your A-disk with the filename of PARTY. Enter:

```
listfile party
```

you should see the following:

```
PARTY SUPPLIES A1
```

Let's request a display of the file, using the TYPE command. Enter:

```
type party supplies
```

You should see the following:

```
BALLOONS
CAKE
PARTY HATS
ICE CREAM
GUESTS
```

Since you really don't need this file, you can erase it from your permanent storage using the ERASE command. Enter:

```
erase party supplies
```

When you receive the ready message (Ready;), you know that the file was erased. Let's check to see if it really was erased. Use the LISTFILE command again to list the files on your A-disk with the filename of PARTY. Since you just erased the file, you'll receive the following message:

```
File not found
Ready(00028);
```

Most CMS commands create or reference disk files, and are as easy to use as the commands shown above. Your CMS disks are among the most important features in your VM/SP virtual machine.

Virtual Disks and How They Are Defined

Under VM/SP, a real direct access storage device (DASD) can be divided into many small areas, called minidisks. Minidisks, often called virtual disks, are defined in the VM/SP directory, as locations on real disks. For CMS applications, you never have to be concerned with the locations of your minidisks; when you use CMS-formatted minidisks, they are, for practical purposes, functionally the same as real disks. Minidisks can also be formatted for use with OS or DOS data sets or VSAM files.

You can have two types of disks, permanent and temporary.

Permanent disks	persist across logons; they are defined in the VM/SP directory entry for your virtual machine.
Temporary disks	are automatically destroyed at logoff. Temporary disks are those you define for your own virtual machine using the CP DEFINE command, or those attached to your virtual machine by the system operator.

Both permanent and temporary disks may be attached to your machine during a terminal session.

Defining Temporary Virtual Disks

Using the CP DEFINE command, you can attach a temporary disk to your virtual machine for the duration of a terminal session. The following command allocates a 10-cylinder temporary disk from a 3330 device and assigns it a virtual address of 291:

```
define t3330 as 291 cyl 10
```

When you define a minidisk, you can choose any valid address that is not already assigned to a device in your virtual machine. Valid addresses for minidisks range from 001 through 5FF.

Formatting Virtual Disks

Before you can use any new virtual disk, you must format it. This applies to new disks that have been assigned to you and to temporary disks that you have allocated with the CP DEFINE command. When you issue the FORMAT command, you must use the virtual address you have defined for the disk and assign a CMS mode letter, for example:

```
format 291 c
```

CMS then prompts you with the following message:

```
DMSFOR603R FORMAT will erase all files on disk C(291).
Do you wish to continue?  Enter 1 (YES) or 0 (NO).
```

You respond:

```
1
```

CMS then asks you to assign a label for the disk, which may be anything you choose. Labels can have a maximum of 6 characters. When the message:

```
DMSFOR605R Enter disk label:
```

is issued, you respond by supplying a disk label. For example, if this is a temporary disk, you might enter:

```
scrtch
```

CMS then erases all the files on that disk, if any existed, and formats the disk for your use and displays the following messages:

```
Formatting disk C
10 cylinders formatted on C(291)
Ready; T=0.15/1.60  11:26:03
```

The FORMAT command should only be used to format CMS disks, that is, disks you are going to use to contain CMS files. In addition, this command allows you a choice of physical disk block size as an option. Refer to the *VM/SP CMS Command Reference* for details.

Sharing Virtual Disks: Linking

Since only one user can own a virtual disk, and there are many occasions that require users to share data or programs, VM/SP allows you to share virtual disks, on either a permanent or temporary basis, by "linking".

Permanent links can be established for you in your VM/SP directory entry. These disks are then a part of your virtual machine configuration every time you log on. You can also have another user's disk temporarily added to your configuration by using the CP LINK command. For example, if you have a program that uses data that resides on a disk identified in userid DATA's configuration as a 194, and you know that the password assigned to this disk is GO, you could issue the command:

```
link to data 194 as 198 r pass= go[1]
```

DATA's 194 disk is then added to your virtual machine configuration at virtual address 198.

The "R" in the command indicates the access mode; in this case, it tells CP that you only want to read files from this disk and you will not be allowed to write on it. If you try to issue this command when someone already has write access to that disk, you will not be able to establish the link. If you want to link to DATA in any event, you can reissue the LINK command using the access mode RR:

```
link data 194 198 rr go[1]
```

The keywords "TO," "AS," and "PASS=" are optional; you do not have to specify them.

[1] The password cannot be entered on the command line if the password suppression facility was specified when your system was installed.

However, note that using the RR access allows one user to read a disk while another is updating the same disk at the same time. This may produce unpredictable results.

You can also use the CP LINK command to link to your own disks. For example, if you log on and discover that another user has access to one of your disks, you may be given read-only access, even if it is a read/write disk. You can request the other user to detach your disk from his virtual machine, and after he has done so, you can establish the link:

```
link * 191 191
```

When you link to your own disks, you can specify the userid as *, and you do not need to specify the access mode or a password.

You can find more information about the CP LINK command and CP access modes in the *VM/SP CP Command Reference.*

Identifying Your Disk To CMS: Accessing

LINK and DEFINE are CP commands: they tell CP to add DASD devices to your virtual machine configuration. CMS must also know about these disks, and you must use the ACCESS command to establish a filemode letter for them:

```
access 194 b
```

CMS uses filemode letters to manage your files during a terminal session. By using the ACCESS command you can control:

- Whether you can write on a disk or only read from it (its read/write status)
- The library search order for programs executing in your virtual machine
- Which disks are to contain the new files that you create.

If you want to know which disks you currently have access to, issue the command:

```
query search
```

You might see the following display:

```
PLC191  191  A    R/W
DAT194  198  B    R/O
CMS190  190  S    R/O
CMS19E  19E  Y/S  R/O
```

The first column indicates the label on the disk (assigned when the disk is formatted), and the second column shows the virtual address assigned to it.

The third column contains the filemode letter. All letters of the alphabet are valid filemode letters.

The fourth column indicates the read/write status of the disk. The 190 and 19E disks in this example are read-only disks that contain the CMS nucleus and disk-resident commands for the CMS system. You will probably use your 191 (A) disk as your primary read/write work disk.

Releasing Virtual Disks

When you no longer need a disk that you linked or temporarily accessed, then release that disk. To release a disk, use the CMS RELEASE command:

```
release c
```

When you want to assign a currently active filemode letter to another disk, issue the ACCESS command to assign that filemode letter to another disk. It is not necessary to release an accessed disk prior to accessing another disk with the same filemode.

When you no longer need disks in your virtual machine configuration, use the CP command DETACH to disconnect them from your virtual machine:

```
detach 194
detach 291
```

If you are going to release and detach the disk at the same time, you can use the DET option of the RELEASE command:

```
release 194 (det
```

When you logoff the disks are released automatically. For more information on controlling disks in CMS, see Chapter 3, "The CMS File System."

Console Output

When you use a 3270 terminal as your virtual machine console, you do not ordinarily retain a console log, as you do on a typewriter terminal. There may be many circumstances in which you need a printed record of your console output, whether it be to obtain a copy of program-generated output, or to retain a record of CP and/or CMS commands that resulted in an error condition. There are two techniques you can use in VM/SP to obtain hard copy representations of display terminal sessions: spooling console output and the 3270 copy function.

Spooling Console Output

The CP SPOOL command provides the CONSOLE operand, which allows you to begin and end console spooling. You enter:

```
spool console start
```

when you want to begin recording your terminal session, and:

```
spool console stop
```

when you have finished. In between, you can periodically close the console file to release for printing whatever has been spooled thus far:

```
spool console close
```

Other operands that you can enter are the same as you might specify for any printer file, such as CLASS, COPY, CONT, and HOLD.

An alternate technique is to spool your console to your own virtual reader:

```
spool console start * class a
```

Then, when you close the console file, instead of being released to the CP printer spool file queue, it is routed to your virtual reader, and you can load it onto your A-disk as a CMS disk file:

```
receive = console file
```

You can then use the editor to examine it (or to delete sections you don't need) and use the PRINT command to obtain a printed copy.

If you are using full-screen CMS, you will need to use logfiles to spool information from your terminal. When you set full-screen on, by default, messages and warnings will be logged for you. Messages will be logged to a file with the filename and filetype of MESSAGE LOGFILE; warnings will be logged to WARNING LOGFILE.

If you wish, you can create a log of your CMS output or other output. To do this, after you are in full-screen CMS, type the command SET LOGFILE CMS ON. Your output will be sent to a file with the filename and filetype of CMS LOGFILE. Later, you could XEDIT or print this file to obtain a hard copy of the work you completed. Refer to the *VM/SP CMS Command Reference* for further information on the SET LOGFILE command.

Copying Your Screen

If you are using a 3270 display terminal, and you have available a 3284, 3286, 3287, 3288, or 3289 printer, you can copy the full screen display currently appearing on the screen. To copy the screen, you have to assign the copying function to a program function key, with the SET command:

```
set pf9 copy
```

Note: The PF key copy function is not available if the printers are dedicated.

Then, whenever you want to copy a screen display, you can press the PF9 key (or whichever key you set). The display is printed on any 3270 display printer that is attached to the same control unit as the display terminal. If, when you press the PF key, the screen status area indicates NOT ACCEPTED, it means that the printer is either not ready or not available. When you press the PF key and receive no response, it means that the screen has been copied.

There is a print matrix available to the 3274 and 3276 user that allows control of the display to printer operations. In addition, a local print key is provided on the display terminal that can be used for copy operations.

Figure 6 is an example of a 3270 screen display that could be copied on the printer. When you use the copy function to copy a screen, all 24 lines of the display screen are copied; the screen status area (indicated as RUNNING in Figure 6) is blank if the 3270 is locally attached. If the 3270 is remotely attached, the entire screen including the screen status area, is copied. You can use the user input area of your screen to key in comments, or your name or userid, if several users are spooling copy files.

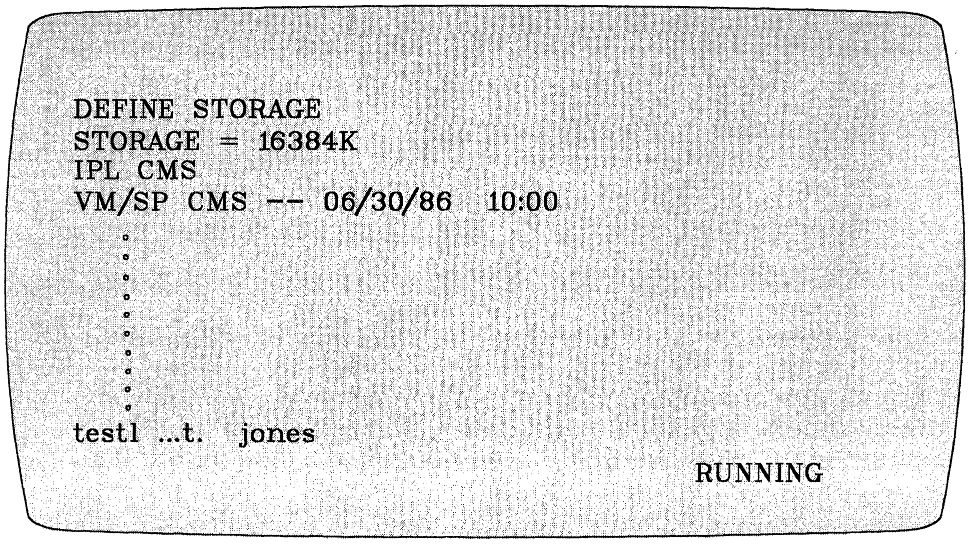

```
DEFINE STORAGE
STORAGE = 16384K
IPL CMS
VM/SP CMS -- 06/30/86  10:00
    .
    .
    .
    .
    .
    .
    .
    .
    .
    .
testl ...t.  jones
                                        RUNNING
```

Figure 6. 3270 Screen Display

For more information about copying screens in XEDIT, refer to the COPYKEY option of the SET PF command in the *VM/SP System Product Editor Command and Macro Reference.*

If you are using full-screen CMS, you can copy your screen with the PUT SCREEN command. This command sends a copy of your physical screen to a CMS file which you can later XEDIT or print. Refer to the *VM/SP CMS Command Reference* for details.

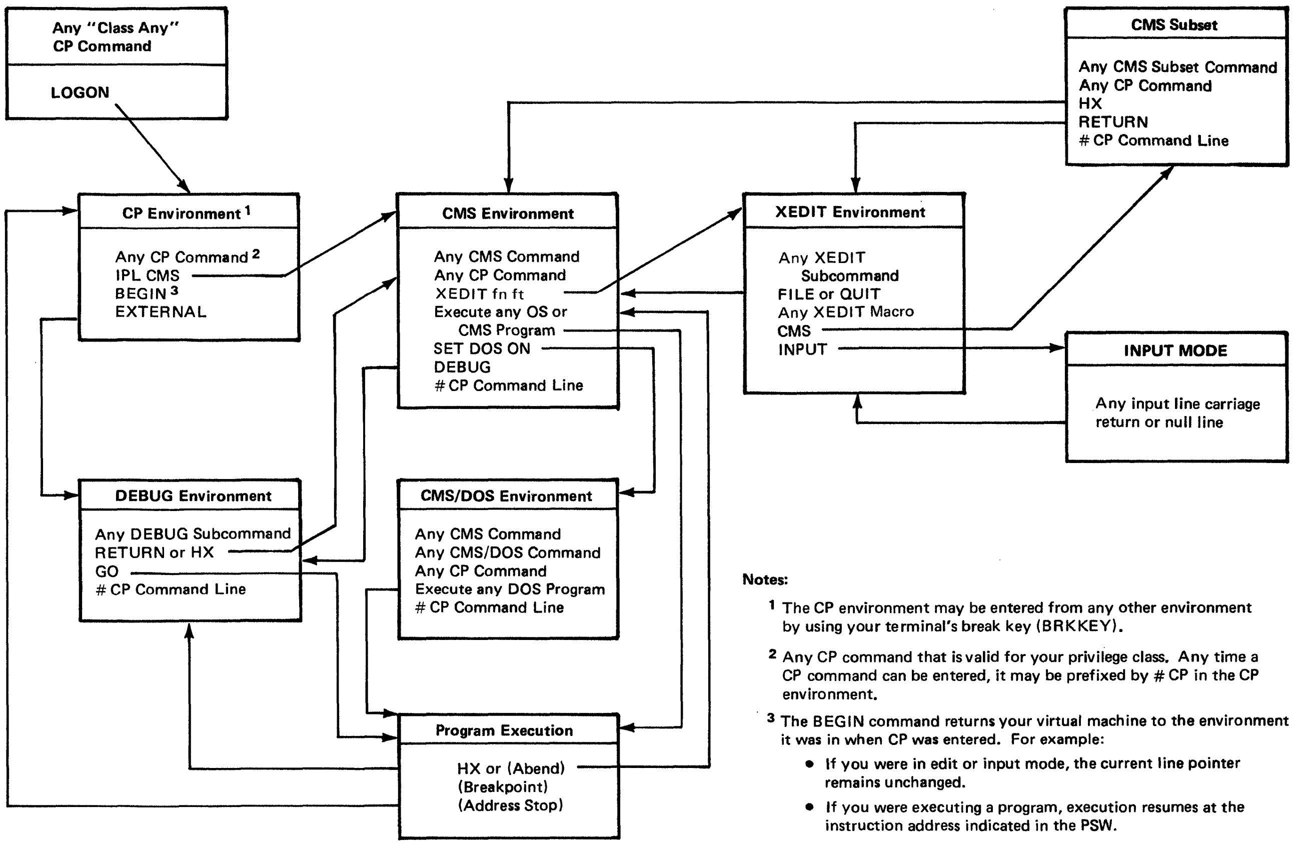
Chapter 2. VM/SP Environments and Mode Switching

When you are using VM/SP, your virtual machine can be in one of two possible "environments," the control program (CP) environment or the virtual machine environment, which may be CMS. The CMS environment has several subenvironments, sometimes called "modes". Each environment or subenvironment accepts particular commands or subcommands, and each environment has its own entry and exit paths, responses and error messages. If you have a good understanding of how the VM/SP environments are related, you can learn to change environments quickly and use your virtual machine efficiently.

This chapter introduces the CP and CMS environments that you use and describes:

- Entry and exit paths
- Command subsets that are valid as input.

Figure 7 on page 32 summarizes the VM/SP command environments and lists the commands and terminal paths that allow you to go from one environment to another.



Notes:

[1] The CP environment may be entered from any other environment by using your terminal's break key (BRKKEY).

[2] Any CP command that is valid for your privilege class. Any time a CP command can be entered, it may be prefixed by # CP in the CP environment.

[3] The BEGIN command returns your virtual machine to the environment it was in when CP was entered. For example:

- If you were in edit or input mode, the current line pointer remains unchanged.
- If you were executing a program, execution resumes at the instruction address indicated in the PSW.

Figure 7. VM/SP Environments and Mode Switching

With the exception of input mode in the edit environment, you can always determine which environment your virtual machine is in by pressing the Return or ENTER key on a null line. The responses you receive and the environments they indicate, are:

Response	Environment
CP	CP
CMS	CMS
CMS (DOS ON)	CMS/DOS
CMS SUBSET	CMS Subset
DEBUG	Debug

The CP Environment

When you log on to VM/SP, your virtual machine is in the CP environment. In this environment, you can enter any CP command that is valid for your privilege class. This publication assumes that you are a general, or class G, user. You can find information about the commands that you can use in the *VM/SP CP Command Reference*.

Only CP commands are valid terminal input in the CP environment. You can, however, preface a CP command with the characters "CP" or "#CP", followed by one or more blanks, although it is not necessary. These functions are described under the section entitled "The CMS Environment" on page 34.

You can enter CP commands from other VM/SP environments. There may be times during your terminal session when you want to enter the CP environment to issue one or more CP commands. You can do this from any other environment by doing either of two things:

1. Issue the command:

   ```
   #cp
   ```

2. Use your terminal's Attention key (PA1 or equivalent). On a 2741 terminal, you must normally press the Attention key twice, quickly, to enter the CP environment.

The following message indicates that your virtual machine is in the CP environment:

```
CP
```

After entering whatever CP commands you need to use, you return your virtual machine to the environment or mode that it came from by using the CP command:

```
begin
```

which, literally, begins execution of your virtual machine.

Note: Once you set full-screen CMS on, the PA1 key no longer serves as an Attention key but performs a windowing function. If you want to use an Attention key to enter the CP environment, you will need to set another key to the attention function. When you set full-screen CMS off, the CP TERMINAL BRKKEY remains as NONE. To reset it to PA1 (the default setting), use the CP TERMINAL command. For further details, refer to the BRKKEY option of the CP TERMINAL command in the *VM/SP CP Command Reference.*

The CMS Environment

You enter the CMS environment from CP by issuing the IPL command, which loads CMS into your virtual storage area. If you are planning to use CMS for your entire terminal session, you should not have to IPL again unless a program failure forces you into the CP environment.

When you issue the IPL command, specify the named system CMS at your installation. For example:

```
ipl cms
```

When your virtual machine is in the CMS environment, you can issue any CMS command and any of the CP commands that are valid for your user privilege class. You can also execute many of your own OS or DOS programs. For further details, refer to *VM/SP CMS for System Programming.*

You can enter CP commands from CMS in any of the following ways:

- Using the implied CP function of CMS (See *Note.*)
- With the CP command
- With the #CP function.

Note: For the most part, you may enter any CP command directly from the CMS environment. This implied CP function is controlled by an operand of the CMS SET command, IMPCP. You can determine whether the implied CP function is in effect for your virtual machine by entering the command:

```
query impcp
```

If the response is:

```
IMPCP      = OFF
```

you can change it by entering:

```
set impcp on
```

When the implied CP function is set off, you must use either the CP command or the #CP function to enter CP commands from the CMS environment. CP commands that you execute in EXEC procedures must always be prefaced by the CP command, regardless of the implied CP setting. An example of using the CP command is:

```
cp close punch
```

When you issue CP commands from the CMS environment either implicitly or with the CP command, you receive, in addition to the CP response (if any), the CMS ready message. If you use the #CP function, discussed next, you do not receive the ready message.

You can preface any CP command with the characters "#CP", followed by one or more blanks. When you enter a CP command this way, the command is processed by CP immediately; it is as if your virtual machine were actually in the CP environment.

Protected Application Environment

A protected application environment provides special enhancements to prevent users with no need to interact with CP from accidentally entering the CP environment.

When you are operating in a protected application environment:

- Pressing the Attention key will not cause you to enter CP mode.
- The terminal break key is set to NONE.
- CP will attempt to initiate an automatic re-IPL upon encountering an error that would cause CMS to abnormally end (abend).

A protected application environment is invoked by using the CONCEAL option of the SET command or the directory OPTION control statement. Your system administrator sets the directory OPTION control statement so that you would be placed in a protected application at logon time. You can use the CONCEAL option of the SET command to invoke a protected application at any time during your terminal session.

To invoke SET CONCEAL, you would simply enter:

```
set conceal on
```

Either way a protected application is invoked, the terminal break key will always be set to NONE. SET CONCEAL OFF will return the break key to its default setting of PA1, for local, remote, and VM/VTAM 3270 graphics terminals.

For further details on the SET CONCEAL command, refer to the *VM/SP CP Command Reference.*

XEDIT and CMS Subset

The System Product Editor is a VM/SP facility that allows you to create and modify data files that reside on CMS disks. The editor environment, more commonly called the edit environment, is entered when you issue the CMS command XEDIT, specifying the identification of a data file you want to create or modify. Complete information about the System Product Editor, invoked via the XEDIT command, is found in the *VM/SP System Product Editor Command and Macro Reference* and the *VM/SP System Product Editor User's Guide.* For introductory tutorial information about editing, refer to the *VM/SP CMS Primer.*

For example:

```
xedit party supplies
```

is the command you would use to enter the edit environment to either create a file called PARTY SUPPLIES or to make changes to a disk file that already exists under that name.

When you enter the edit environment, your virtual machine is automatically in edit mode, where you can issue XEDIT subcommands, CMS commands, or CP commands. After you enter the XEDIT subcommand:

```
input
```

data lines that you enter are considered input to the file. To return to edit mode, you must enter a null line.

If you issue the XEDIT subcommand:

```
cms
```

the editor responds:

```
CMS subset
```

and your virtual machine is in CMS subset mode. When in CMS subset mode, you can issue any valid CMS subset command, that is, a CMS command that is allowed in CMS subset mode. The commands that are not allowed in the CMS subset environment are commands that execute in the user area. You can also issue CP commands. To return to edit mode, you use the special CMS subset command, RETURN. If you enter the Immediate command HX, your editing session terminates abnormally and your virtual machine returns to the CMS environment.

For more information on CMS subset, refer to *VM/SP CMS for System Programming.*

When you are finished with an edit session, you return to the CMS environment by issuing the FILE subcommand, which indicates that all modifications or data insertions that you have made should be written onto a CMS disk. Otherwise, you can issue the subcommand QUIT, which tells the editor not to save any modifications or insertions made since the last time the file was written.

DEBUG

CMS DEBUG is a special CMS facility that provides subcommands to help you debug programs at your terminal. Your virtual machine enters the debug environment when you issue the CMS command:

```
debug
```

You may want to enter this command after you have loaded a program into storage and before you begin executing it. At this time you can set "breakpoints," or address stops, where you wish to halt your program's execution so that you can examine and change the contents of general registers and storage areas. When these breakpoints are encountered, your virtual machine is placed in the debug environment. You can also enter the debug environment by issuing the CP EXTERNAL command, which causes an external interrupt to your virtual machine.

Valid DEBUG subcommands that you can enter in this environment are:

BREAK	GO	RETURN
CAW	GPR	SET
CSW	HX	STORE
DEFINE	ORIGIN	X
DUMP	PSW	

You can also use the #CP function in the debug environment to enter CP commands.

You leave the debug environment in any of the following ways:

- If the program you are running completes execution, you are returned to the CMS environment.

- If your virtual machine entered the debug environment after a breakpoint was encountered, it returns to CMS when you issue the DEBUG subcommand:

  ```
  hx
  ```

  To continue the execution of your program, you use the DEBUG subcommand:

  ```
  go
  ```

- If your virtual machine is in the debug environment and is not executing a program, the DEBUG subcommand:

  ```
  hx
  ```

  returns it to the CMS environment.

Refer to *VM Diagnosis Guide* for more information on using DEBUG.

CMS/DOS

If you are a VSE user, the CMS/DOS environment provides you with all the CMS interactive functions and facilities, as well as special CMS/DOS commands which simulate VSE functions. The CMS/DOS environment becomes active when you issue the command:

```
set dos on
```

When your virtual machine is in the CMS/DOS environment you can issue any command that would be valid in the CMS environment, including the facilities of XEDIT, DEBUG, and EXEC, except for CMS commands or program modules that load and/or execute programs that use OS macros or functions.

The following commands are provided in CMS/DOS to test and develop DOS programs, and to provide access to VSE libraries:

ASSGN	DOSPLI	LISTIO
CATCHECK	DSERV	OPTION
DLBL	ESERV	PSERV
DOSLIB	FETCH	RSERV
DOSLKED	FCOBOL	SSERV

Your virtual machine leaves the CMS/DOS environment when you issue the command:

```
set dos off
```

If you reload CMS (with an IPL command) during a terminal session, you must also reissue the SET DOS ON command to return to the CMS/DOS environment. For more information about the CMS/DOS environment, see *VM/SP CMS for System Programming.*

Interrupting Program Execution

When you are executing a program under CMS or executing a CMS command, your virtual machine is not available for you to enter commands. There are, however, ways in which you can interrupt a program and halt its execution, either temporarily, in which case you can resume its execution, or permanently, in which case your virtual machine returns to the CMS environment. In both cases, you interrupt execution by creating an "attention interruption," which may take two forms:

- An attention interruption to your virtual machine operating system
- An attention interruption to the control program.

Using the Attention Key

Attention interrupts result in what are known as virtual machine (VM) or control program (CP) "reads" being presented to your virtual console. The two keys on your 3270 keyboard that signal interruptions are the PA1 key -- REQ key on a 3278 Model 2A -- and the ENTER key. Throughout this publication, interruption signaling has been described in terms of the Attention key.

You can enter the CP environment by pressing the PA1 key. Whenever you press this key, your virtual machine is placed in a CP READ status, and you can enter any CP command. From the CP environment, you must use the CP command BEGIN to resume execution of your virtual machine. On a typewriter terminal, the keyboard unlocks when a read occurs.

Whether you have to press the Attention key once or twice depends on the terminal mode setting in effect for your virtual machine. This setting is controlled by the CP TERMINAL command:

```
terminal mode vm
```

The setting VM is the default for virtual machines; you do not need to specify it. The VM setting indicates that one depression of the Attention key sends an interruption to your virtual machine, and that two depressions results in an interruption to the control program (CP).

The CP setting for terminal mode, which is the default for the system operator, indicates that one depression of the Attention key results in an interruption to the control program (CP). If you are using your virtual machine to run an operating system other than CMS, you might wish to use this setting. Issue the command:

```
terminal mode cp
```

Note: Once you set full-screen CMS on, the PA1 key no longer serves as an Attention key but performs a windowing function. If you want use a break key, you will need to set another key to the attention function. When you set full-screen CMS off, the CP TERMINAL BRKKEY remains as NONE. To reset it to PA1 (the default setting), use the CP TERMINAL command.

Interrupting Your Programs

On a typewriter terminal, the Attention key, pressed once, causes a virtual machine interruption (if the terminal mode is set to VM); you must use it when you want to enter an Immediate command, such as HT or HX. On a display terminal, you can enter these commands whenever your virtual machine is in a running status, without having to signal an interruption before you enter the command.

Sometimes, however, if your terminal is displaying output very rapidly, you must wait until the screen is full and the screen status area indicates a MORE... status before you attempt to enter the HT or HX command.

The ENTER key can also be used as an interruption signaling key. If you press it once when your virtual machine is running, you will place your virtual machine in the VM READ status, so you can enter a command.

While in VM READ, you can pass null lines or Immediate commands to CMS. This procedure is particularly useful when you wish to stop execution of an EXEC that is issuing VM READs to the terminal. You can type HI or HX and have it passed to CMS without being read and interpreted as data by the EXEC. At the VM READ, use the cursor movement key to move the cursor back one space from its current position at the command line. (This results in the cursor being positioned in the lower right corner of the screen, three lines up from the bottom). Press the ENTER key. The cursor will return to the command line, and your terminal will remain in VM READ status. At this point, you can enter HI, HX, or any Immediate command, or if you wish to resume execution of the EXEC, enter the next line of data.

Note: If you are working in full-screen CMS and wish to interrupt execution of an EXEC reading data from your terminal, type any Immediate command prefixed by '#'. Since '#' is the default linend character, the command will be immediately interpreted, and execution will be interrupted. For example, to halt execution of an EXEC, you would type #HX.

Virtual Machine Interruptions

While a command or program is executing, if you press the ENTER key on a 3270 (or the Attention key once on a 2741), you have created a virtual machine interruption.

Halting Screen Displays

When your terminal is displaying successive screens of output from a program or a CMS command, you can use the HT or HX Immediate commands to halt the display or the execution of the command, respectively. If your terminal is writing the information at a very rapid rate, you may have difficulty entering the HT or HX command. In these circumstances, press the ENTER key twice to force your terminal to a CP READ status. Then, you can use the CP command ATTN or REQUEST to signal a virtual machine read. When the screen status area indicates VM READ, you can enter HX or HT. The program halts execution, your terminal will accept an input line, and you may:

- Terminate the execution of the program by issuing an Immediate command to halt execution:

  ```
  hx
  ```

  The HX command causes the program to abnormally terminate (abend).

- Enter a CMS command. The command is stacked in a console stack and is processed by CMS when your program is finished executing and the next virtual machine read occurs. For example:

  ```
  print abc listing
  ```

  After you enter this line, the program resumes execution.

- If the program is directing output to your terminal and you wish only to halt the terminal display, use the Immediate command:

  ```
  ht
  ```

  The program resumes execution. Terminal output can also be suppressed immediately when you enter a command by placing #HT after the command. For example, you would enter:

  ```
  tape dump #ht
  ```

  to suppress output from the TAPE DUMP command. The logical line end character (#) allows the Immediate command HT to be accepted; program execution proceeds without typing.

  You can, if you want, cause another interruption and request that typing be resumed by entering the RT (resume typing) command:

  ```
  rt
  ```

- Enter a null line; your program continues execution. The null line is stacked in the console stack and read by CMS as a stacked command.

HX, HT, and RT are three of the CMS Immediate commands. They are "immediate" because they are executed as soon as they are entered. Unlike other commands, they are not stacked in the console stack. You can only enter an Immediate command following an attention interruption.

Control Program Interruptions

You can interrupt a program and enter the CP environment directly by pressing the PA1 key on a 3270 or by pressing the Attention key twice, quickly, on a 2741. Then, you can enter any CP command. To resume the program's execution, issue the CP command:

```
begin
```

If your terminal is operating with the terminal mode set to CP, pressing the Attention key once places your virtual machine in the CP environment.

Note: Remember that in full-screen CMS mode, PA1 will pop the WM window. If you wish to override this default setting, you can do so by using the BRKKEY option of the CP TERMINAL command. When you set full-screen CMS off, the CP TERMINAL BRKKEY remains as NONE. To reset it to PA1 (the default setting), use the CP TERMINAL command. For more information, refer to the *VM/SP CP Command Reference.*

Address Stops and Breakpoints

You can halt program execution after an instruction at a particular address with the CP PER command. Entering the command:

```
per i r 201ea
```

causes program execution to halt after the instruction at location X'201EA'. A program may also be interrupted by an instruction address stop, which you specifically set by the CP command ADSTOP. For example, if you issue the command:

```
adstop 201ea
```

an address stop is set at virtual storage location X'201EA'. When your program reaches this address during its execution, it is interrupted and your virtual machine is placed in the CP environment, where you can issue any CP command, including another ADSTOP command, before resuming your program's execution with the CP command BEGIN.

Using APL

If you have a 3277 or 3278 display station equipped with an APL keyboard, you can use APL on a 3270 terminal in CMS. You invoke the APL virtual machine by issuing the command specified in the VS APL Program Product documentation. This command invokes the VS APL-CMS interface program. You are then prompted to press the APL On/Off key which is on your terminal; pressing this key changes the keyboard to APL character input mode. You are then prompted to press the ENTER key to continue.

EBCDIC or APL characters can always be displayed; the APL On/Off key does not change this. The VS APL-CMS interface program issues the TERMINAL APL ON command for you and selects the appropriate translation tables. The TERMINAL APL ON command automatically forces a TERMINAL TEXT OFF condition. The interface program then invokes the VSAPL program. When the VSAPL ready message appears on the screen, you can use APL.

You can send a copy of your display screen to a locally or remotely attached printer. Be sure that the printer you send your output to has the APL feature installed; if it does not, the APL characters are not printed. Most system printers do not have an APL print chain; therefore you may need to use the copy function to direct your screen output displays to a 3284, 3286, or 3287 printer.

For more information on using APL, refer to the SET APL command in the *VM/SP CMS Command Reference.* You may also find it helpful to refer to the *VM/SP System Product Editor Command and Macro Reference* for information on using the XEDIT command SET APL.

Error Situations

If you do not have the APL hardware feature installed on your 3277 or 3278 but you invoke APL:

- The VSAPL program is invoked and the TERMINAL APL ON command is issued.
- You cannot communicate with the VSAPL program.
- Any APL characters that are written to the screen appear as blanks.

  If you have the APL feature installed on your terminal, but invoke APL manually without issuing the TERMINAL APL ON command or issue TERMINAL APL OFF at sometime during APL processing:

- The VSAPL program is activated.
- You cannot communicate with the VSAPL program.
- Any APL characters written to the screen appear as blanks.

If you attempt to use the APL O/S (overstrike) key when the APL hardware key is set off, it acts as a backtab key and repositions the cursor to the beginning of the user input area.

Leaving the APL Environment

Issue the APL command:

```
)OFF
```

to log off VM/SP.

Issue the APL command:

```
)OFF HOLD
```

to return to CMS. This APL command invokes the VS APL-CMS interface program, which:

- Issues the TERMINAL APL OFF command
- Prompts you to press the APL hardware key
- Returns to CMS.

Note: The APL hardware feature is a key, not a switch. Each time you press the APL key you reverse its on/off setting. To determine whether APL is on or off, press a key that represents a special APL character. If the character displayed is an APL character, the hardware APL feature is set on. If the character displayed is a non-APL character, you must press the APL key once to set the APL feature on.

Using the 3270 Text Feature

If you have a 3277 or 3278 display station equipped with the Data Analysis Text keyboard, you can key in, as well as display, all of the special text characters. For a description of these characters, see the *VM/SP Terminal Reference.* These characters are in addition to those available with standard EBCDIC 3270 terminals. If you have a suitably equipped printer attached to your 3270, you can use the SET PFnn COPY function to obtain a printed copy of the screen.

Note: This procedure for obtaining a printed copy of your screen pertains to sessions when you *are not* using full-screen CMS. Refer to "Copying Your Screen" on page 29 for information on how to copy your screen in full-screen CMS.

When you want to activate the text feature, and use the special characters, enter the command:

```
terminal text on
```

The TERMINAL TEXT ON command automatically forces the TERMINAL APL OFF command. Now, you can use any of the special characters when you enter, change, or locate text lines in a file.

You leave the special text environment by entering the command:

```
terminal text off
```

Refer to the *VM/SP CMS Command Reference* and *VM/SP System Product Editor Command and Macro Reference* for further information on using the SET TEXT commands to select appropriate translation tables for special characters.

Error Situations

If you do not have the appropriate text hardware feature on your 3270, but attempt to display a file that contains the characters, the characters appear as blanks on a 3277, and as hyphens on a 3276 and a 3278.

If you inadvertently issue the TERMINAL TEXT ON command while using a terminal that does not have the text capability, you must do the following to return to normal operating procedures:

1. Press the PA1 key to enter the CP environment.
2. Key in, in uppercase letters only, the command:

```
TERMINAL TEXT OFF
```

Notes:

1. *The 3270 text hardware feature is activated by a key, not a switch. Each time you press the TEXT On/Off key, you reverse its setting. If your terminal has a red light on the text keyboard, it will be illuminated when the text feature is on.*

2. *Compound characters, such as a character/-backspace/-character combination, are still entered and displayed as three characters. The screen position occupied by the backspace character appears as a blank because the character (X'16') is nondisplayable. For information on displaying nondisplayable characters, refer to the description of the NONDISP commands in the VM/SP CMS Command Reference and VM/SP System Product Editor Command and Macro Reference.*

Chapter 3. The CMS File System

The file is the essential unit of data in the CMS system. CMS disk files are unique to the CMS system and cannot be read or written using other operating systems. When you create a file in CMS, you name it using a file identifier. The file identifier consists of three fields:

- `Filename (fn)`
- `Filetype (ft)`
- `Filemode (fm)`

When you use CMS commands and programs to modify, update, or reference files, you must identify the file by using these fields. Some CMS commands require you to enter only the filename, or the filename and filetype; others require you to enter the filemode field as well. This chapter contains information about the things you must consider when you give your CMS files their identifiers, notes on the file system commands that create and modify CMS files, and additional notes on using CMS disks.

CMS File Formats

The CMS file management routines write CMS files on disk in fixed physical blocks, regardless of whether they have fixed- or variable-length records. For most of your CMS applications, you never need to specify either a logical record length and record format or block size when you create a CMS file.

When you create a file using one of the CMS editors, the file has certain default characteristics, based on its filetype. The special filetypes recognized by the editor, and their applications, are discussed in the section entitled "What Are Reserved Filetypes?" on page 50.

How CMS Files Get Their Names

When you create a CMS file, you can give it any filename and filetype you wish. The rules for forming filenames and filetypes are:

- The filename and filetype can each be from one to eight characters.

- The valid characters are A-Z, a-z, 0-9, $, #, @, +, - (hyphen), : (colon), and _ (underscore).

Note: Lowercase letters within a fileid are valid for use within the CMS file system. However, some CMS commands do not support fileids that contain lowercase letters.

When you enter a command into the VM/SP system, VM/SP translates your input line by either the user defined input table or by the uppercase table. See the CMS SET INPUT command in the *VM/SP CMS Command Reference*. If you do not have an input table, you can just enter the command in lowercase and VM/SP translates your input line into uppercase characters.

Note: When defining input characters be sure that you will not end up with a fileid containing invalid characters.

The # and @ characters are line editing symbols in VM/SP; when you use them to identify a file, you must precede them with the logical escape symbol ("). See Appendix A, "Considerations for Line Mode Terminals" for a list of logical line editing symbols.

The third field in the file identifier, the filemode, indicates the mode letter (A-Z) currently assigned to the virtual disk on which you want the file to reside. When you use the editor to create a file, and you do not specify this field, the file you create is written on your A-disk, and has a filemode letter of A.

The filemode letter, for any file, can change during a terminal session. For example, when you log on, your virtual disk at address 191 is accessed as your A-disk, so a file on that disk named SPECIAL EVENTS has a file identifier of:

```
SPECIAL EVENTS A
```

If, however, you later access another disk as your A-disk, and access your 191 as your B-disk, then this file has a file identifier of:

```
SPECIAL EVENTS B
```

Duplicate Filenames or Filetypes

You can give the same filename to as many files on a given disk as you want, as long as you assign them different filetypes. Or you can create many files with the same filetype but different filenames.

For the most part, filenames that you choose for your files have no special significance to CMS. If, however, you choose a name that is the same as the name of a CMS command, and the file that you assign this name to is an executable module or EXEC procedure, then you may encounter difficulty if you try to execute the CMS command whose name you duplicated.

For an explanation of how CMS identifies a command name, see the section entitled "CMS Command Search Order" later in this chapter.

Many CMS commands allow you to specify one or more of the fields in a file identifier as an asterisk (*) or equal sign (=), which identify files with similar fileids.

Using Asterisks (*) in Fileids

Some CMS commands that manipulate disk files allow you to enter the filename and/or filetype fields as an asterisk (*), indicating that all files of the specified filename/filetype are to be modified. These commands are:

```
COPYFILE    RENAME
ERASE       TAPE DUMP
```

For example, if you specify:

```
erase * test a
```

all files with a filetype of TEST on your A-disk are erased. The LISTFILE command allows you to request similar lists. If you specify an asterisk for a filename or filetype, all of the files of that filename or filetype are listed. There is an additional feature that you can use with the LISTFILE command, to obtain a list of all the files that have a filename or filetype that begin with the same character string. For example:

```
listfile t* assemble
```

produces a list of all files on your A-disk with filenames beginning with the letter T and with the filetype of assemble. The command:

```
listfile tr* a*
```

produces a list of all files on your A-disk with filenames beginning with the letters TR and with filetypes beginning with the letter A.

Equal Signs in Output Fileids

The COMPARE, COPYFILE, RENAME, and SORT commands allow you to enter output file identifiers as equal signs (=), to indicate that it is the same as the corresponding input file identifier. For example:

```
copyfile myprog assemble b = = a
```

copies the file MYPROG ASSEMBLE from your B-disk to your A-disk, and uses the same filename and filetype as specified in the input fileid for those positions in the output fileid.

Similarly, if you enter the command:

```
rename temp * b perm = =
```

all files with a filename of TEMP are renamed to have filenames of PERM; the existing filetypes of the files remain unchanged.

What Are Reserved Filetypes?

For the purposes of most CMS commands, the filetype field is used merely as an identifier. Some filetypes, though, have special uses in CMS; these are known as "reserved filetypes".

Nothing prevents you from assigning any of the reserved filetypes to files that are not being used for the specific CMS function normally associated with that filetype.

Some reserved filetypes also have special significance to the System Product Editor. When you use the XEDIT command to create a file with a reserved filetype, the editor assumes various default characteristics for the file, such as record length and format, tab settings, translation to uppercase, truncation column, and so on.

Filetypes for CMS Commands

Reserved filetypes sometimes indicate how the file is used in the CMS system: the filetype ASSEMBLE, for example, indicates that the file is to be used as input to the assembler; the filetype TEXT indicates that the file is in relocatable object form, and so on. Many CMS commands assume input files of particular filetypes, and require you to enter only the filename on the command line. For example, if you enter:

```
synonym test
```

CMS searches for a file with a filetype of SYNONYM and a filename of TEST. A file named TEST that has any other filetype is ignored.

Some CMS commands create files of particular filetypes, using the filename you enter on the command line. The language processors do this as well; if you are recompiling a source file, but wish to save previous output files, you should rename them before executing the command.

Figure 8 on page 51 lists the filetypes used by CMS commands and describes how they are used.

In addition to these CMS filetypes, there are special filetypes reserved for use by the language processors, which are IBM licensed programs. These filetypes, and the commands that use them, are:

Filetypes	Commands
COBOL, SYMDMP, TESTCOB	COBOL, FCOBOL, TESTCOB
FORTRAN, FREEFORT, FTnn001, TESTFORT	FORTRAN, FORTGI, FORTHX, GOFORT, TESTFORT
PLI, PLIOPT	DOSPLI, PLIC, PLICR, PLIOPT
RPGII	RPGII
VSBASIC, VSBDATA	VSBASIC

For details on how to use these filetypes, consult the documentation for the appropriate licensed program.

Filetype	Command	Comments
AMSERV	AMSERV	Contains VSAM access method services control statements executed with the AMSERV command.
ASM3705	ASM3705 GEN3705	Used by system programmers to generate the 3704/3705 control program.
ASSEMBLE	ASSEMBLE	Contains source statements for assembler language programs.
AUXxxxx	UPDATE	Points to files that contain UPDATE control statements for multiple updates.
CNTRL	UPDATE	Lists files that either contain UPDATE control statements or point to additional files.
COPY	MACLIB	Can contain COPY control statements and macros or copy files to be added to MACLIBs.
DIRECT	DIRECT	Contains entries for the VM/SP user directory file. The system programmer controls this file.
DCSSMAP	LANGGEN	Shows the location of each application text deck that was loaded into the DCSS.
EXEC	EXEC GEN3705 LISTFILE	Can contain sequences of CMS or user-written commands, with execution control statements.
EXPAND	EXPAND	Contains a list of control sections and the size of the expansion.
GCS	EXEC	Can contain sequences of GCS or user-written commands, with execution control statements.
GLOBALV	DEFAULTS GLOBALV	Contains variables used by GLOBALV.
GROUP	GROUP	Contains entries for the following data blocks used to describe a GCS virtual machine group: CONFIGURATION, SEGMENT, and AUTHUSERS.

Figure 8 (Part 1 of 5). Filetypes Used by CMS Commands

Filetype	Command	Comments
HELPABBR	HELP	Contains descriptive information for CP, CMS, IPCS, and Transparent Services Access Facility (TSAF) commands, messages, Restructured Extended Executor (REXX), EXEC 2, and EXEC statements, CMS editor and System Product Editor subcommands, and menu, task, and command abbreviation lists and the SQL/Data System Program Product (5748-XXJ) (only if you have this installed on your system.)
HELPCMS		
HELPCMSQ		
HELPCMSS		
HELPCP		
HELPCPQU		
HELPCPSE		
HELPDEBU		
HELPEDIT		
HELPEXC2		
HELPEXEC		
HELPGROU		
HELPHELP		
HELPIPCS		
HELPMENU		
HELPMSG		
HELPPREF		
HELPQUER		
HELPREXX		
HELPSET		
HELPSQLD		
HELPTASK		
HELPTSAF		
HELPXEDI		
$HLPABBR	HELPCONV	Contains descriptive information for CP, CMS, IPCS, and Transparent Access Services Facility (TSAF) commands, messages, Restructured Extended Executor (REXX), EXEC 2, and EXEC statements, CMS editor and System Product Editor subcommands, and menu, task, and command abbreviation lists and the SQL/Data System Program Product (5748-XXJ) (only if you have this installed on your system.)
$HLPCMS		
$HLPCMSQ		
$HLPCMSS		
$HLPCP		
$HLPCPQU		
$HLPCPSE		
$HLPDEBU		
$HLPEDIT		
$HLPEXC2		
$HLPEXEC		
$HLPGROU		
$HLPHELP		
$HLPIPCS		
$HLPMENU		
$HLPMSG		
$HLPPREF		
$HLPQUER		
$HLPREXX		
$HLPSET		
$HLPSQLD		
$HLPTASK		
$HLPTSAF		
$HLPXEDI		
LANGGCTL	LANGGEN	The control file used by LANGGEN.

Figure 8 (Part 2 of 5). Filetypes Used by CMS Commands

Filetype	Command	Comments
LANGMAP	LANGMERG	Shows where language information is stored within a text deck.
LANGMCTL	LANGMERG	The control file used by LANGMERG.
LIST38PP	PRINT	Is a file which is suitable for printing on a 3800 Model 3 page printer. The PRINT command will assume that the file contains a carriage control character in column one and will also assume the OVERSIZE option.
LIST3800	PRINT	Is a file which is suitable for printing on a 3800 printer. The PRINT command will assume that the file contains a carriage control character in column one and a TRC (Table Reference Character) in column two.
LIST3820	PRINT	Is a file which is suitable for printing on a 3820 page printer. The PRINT command will assume that the file contains a carriage control character in column one and will also assume the OVERSIZE option.
LISTCPDS	PRINT	Is a file which is suitable for printing on a page printer. The PRINT command will assume that the file contains a carriage control character in column one and will also assume the OVERSIZE option.
LISTING	AMSERV ASSEMBLE ASM3705 COBOL DOSPLI FCOBOL GENMSG LOADLIB PLIOPT PRINT	Listings are produced by the assembler, the language processors, and the AMSERV, GENMSG, and LOADLIB commands.
LKEDIT	LKED	Contains the printer output from the LINK EDIT of a CMS text file or OS object module.
LOADLIB	FILEDEF GLOBAL LKED LOADLIB NUCXLOAD OSRUN QUERY ZAP	Is a library created by the LKED command or the LOADLIB utility command. The GLOBAL or FILEDEF command identifies the libraries that should be searched for program execution. NUCXLOAD loads a member of a CMS LOADLIB library or an OS module library. OSRUN executes a member of a CMS LOADLIB library or an OS module library. Query indicates the libraries that were affected by the GLOBAL command. ZAP is used to modify an existing LOADLIB member.
LOGFILE	SET LOGFILE	Contains a log of information written to a virtual screen.

Figure 8 (Part 3 of 5). Filetypes Used by CMS Commands

Filetype	Command	Comments
MACLIB	GLOBAL MACLIB MACLIST	Library members contain macro definitions or copy files; the MACLIB command creates the library, and lists, adds, deletes, or replaces members. The GLOBAL command identifies which macro libraries should be searched during an assembly or compilation. The MACLIST command lists all members in a specified MACLIB, with the ability to edit and issue commands from the list.
MACRO	MACLIB	Contains macro definitions to be added to a CMS macro library (MACLIB).
MAP	INCLUDE LOAD MACLIB TAPE TXTLIB	Maps created by the LOAD and INCLUDE commands indicate entry point locations; the MACLIB, TXTLIB, and TAPE commands produce MAP files.
MODULE	GENMOD LOADMOD MODMAP NUCXLOAD	MODULE files created by the GENMOD command are nonrelocatable executable programs. The LOADMOD command loads a MODULE file for execution; the MODMAP command displays a map of entry point locations. NUCXLOAD loads a module into free storage and defines it as a nucleus extension.
NAMES	NAMEFIND NAMES	Contains information regarding users with whom you communicate.
NETLOG	RECEIVE SENDFILE	Contains records which log the transmission of files sent by or received by you.
NOTEBOOK	RECEIVE SENDFILE	Contains notes sent to you or sent by you to other users.
SYNONYM	SYNONYM	Contains a table of synonyms for CMS commands and user-written EXEC and MODULE files.
TEXT	ASSEMBLE INCLUDE LOAD TXTLIB	TEXT files contain relocatable object code created by the assembler and compilers. The LOAD and INCLUDE commands load them into storage for execution. The TXTLIB command manipulates libraries of TEXT files.
TXTLIB	GLOBAL TXTLIB	Library members contain relocatable object code. The TXTLIB command creates the library, and lists or deletes existing members. The GLOBAL command identifies TXTLIBs to search.
TXTxxxxx	GENMSG CONVERT COMMANDS	Contains the object code for language files (message repositories).
UPDATE	UPDATE	Contains UPDATE control statements for single updates applied to source programs.
UPDLOG	UPDATE	Contains a record of additions, deletions, or changes made with the UPDATE command.

Figure 8 (Part 4 of 5). Filetypes Used by CMS Commands

Filetype	Command	Comments
UPDTxxxx	UPDATE	Contains UPDATE control statements for multilevel updates.
ZAP	ZAP ZAPTEXT	Contains control records for the ZAP and ZAPTEXT commands, which are used by system support personnel.

Figure 8 (Part 5 of 5). Filetypes Used by CMS Commands

Output Files: TEXT and LISTING

Output files from the assembler and the language processors are logically related to the source programs by their filenames. Some of these files are permanent and some are temporary. For example, if you issue the command:

```
assemble myfile
```

CMS locates a file named MYFILE with a filetype of ASSEMBLE and the system assembler assembles it. If the file is on your A-disk, then when the assembler completes execution, the permanent files you have are:

```
MYFILE ASSEMBLE A1
MYFILE TEXT     A1
MYFILE LISTING  A1
```

where the TEXT file contains the object code resulting from the assembly, and the LISTING file contains the program listing generated by the assembly. If any TEXT or LISTING file with the same name previously existed, it is erased. The source input file, MYFILE ASSEMBLE A1, is neither erased nor changed.

Because these files are CMS files, you can use the editor to examine or modify their contents. If you want a printed copy of a LISTING file, you can use the PRINT command to print it. If you want to examine a TEXT file, you can use the TYPE or PRINT command specifying the HEX option.

Note: If a TEXT file contains control changes for the terminal, the edit lines may not be displayed in their true form. Therefore, it is suggested you do not use the editor for TEXT files because the results are unpredictable. Instead, use the TYPE or PRINT commands with the HEX option to display TEXT decks. Use ZAPTEXT to modify the TEXT decks.

Filetypes for Temporary Files

The filetypes of files created by the assembler and language processors for use as temporary workfiles are:

SYSUT1	SYS001	SYS004
SYSUT2	SYS002	SYS005
SYSUT3	SYS003	SYS006
SYSUT4		

Figure 9. Filetypes for Temporary Work Files

CMS handles all SYSUTx and SYS00x files as temporary files.

The CMS AMSERV command, executing VSAM utility functions, uses two workfiles that have filetypes of LDTFDI1 and LDTFDI2.

The CMS RECEIVE command, when receiving files in DISK DUMP format, creates a temporary file with the fileid $A$A$A$A $B$B$B$B A.

| The CMS SET LANGUAGE command creates a temporary file with a filename of SET$TEMP and a filetype of TEXT. This file is created when making user language additions to the message repository, user parser table, or synonym table.

Disk space is allocated for temporary files on an as-needed basis. They are erased when processing is complete. If a program you are executing is terminated before completion, these workfiles may remain on your disk. You can erase them.

CMSUT1 Files

The CMSUT1 filename or filetype is used by CMS commands that create files on your CMS disks. The CMSUT1 file is used as a workfile and is erased when processing is complete. When a command fails to complete execution properly, the CMSUT1 file may not be erased. CMSUT1 files are reserved for system usage, and use of these files may cause unpredictable results. The commands, and the filenames or filetypes they assign to files they create, are listed below.

Command	Filename
CONVERT COMMANDS	COMMANDS
COPYFILE	COPYFILE
DISK LOAD	DISK
EDIT	EDIT
EXPAND	CMSUT1 (filetype=TEXT)
HELP	HELP
HELPCONV	HELPCONV
INCLUDE	DMSLDR
LANGMERG	'Application ID'NLS
LOAD	DMSLDR

MACLIB	DMSLBM
READCARD	READCARD
TAPE LOAD	TAPE
UPDATE	fn (the filename of the UPDATE file)
XEDIT	XEDTEMP
ZAPTEXT	CMSUT1(filetype=TXTLIB)

Note: CONVERT COMMANDS also produces temporary CMSUT2 files with a filename of COMMANDS. LANGMERG produces temporary CMSUT2 files with the a filename composed of the Application ID followed by a 1- to 5-character Language ID.

Filetypes for Documentation

There are two CMS reserved filetypes for which the System Product Editor accepts (by default) uppercase and lowercase input data. These are MEMO and SCRIPT.

- You can use MEMO files to document program notes or to write reports.
- The SCRIPT filetype is used by the SCRIPT command. This command invokes a text processor that is part of the IBM Document Composition Facility program product.

Filemode Letters and Numbers

The filemode field of a CMS file identifier has two characters: the filemode letter and the filemode number.

- The filemode letter is established by the ACCESS command and specifies the virtual disk on which a file resides: A through Z.
- The filemode number is a number from 0 to 6, which you can assign to the file when you create it or rename it; if you do not specify it, the value defaults to 1.

How you access your disks and what filemode letters you give them with the ACCESS command depends on how you want to use the files that are on them.

For most of the reading and writing you do of files, you use your A-disk, which is also known as your primary disk. This is a read/write disk. You may access other disks in your configuration, or access linked-to disks, in read-only or read/write status, depending on whether you have a read-only or read/write link.

When you load CMS (with the IPL command), your virtual disk at address 191 is accessed for you as your A-disk. Your virtual disk at address 190 (the system disk) is accessed as your S-disk; and the disk at 19E is accessed as an extension of your S-disk, with a mode letter of Y. The S-disk and Y-disk are accessed for only mode S2 and Y2 files, thus:

```
access 190 S/S * * S2
access 19E Y/S * * Y2
```

In addition, if you have a disk defined at address 192, it is accessed for you as your D-disk. If the 192 disk has not been formatted, CMS will do it automatically and label the minidisk SCRTCH.

If ACCESS is the first command issued after an IPL of the CMS system, only the A-disk is not automatically defined. Another ACCESS command must be issued to define the A-disk.

The actual letters you assign to any other disks (and you may reassign the letters A, D, and Y), is arbitrary; but it does determine the CMS search order, which is the order in which CMS searches your disks when it is looking for a file. The order of search (when all disks are being searched) is alphabetical: A through Z. If you have duplicate file identifiers on different disks, you should check your disk search order before issuing commands against that filename to be sure that you will get the file you want. You can find out the current search order for your virtual disks by issuing the command:

```
query search
```

You can also access disks as logical extensions of other disks, for example:

```
access 235 b/a
```

The "/A" indicates that the B-disk is to be a read-only extension of the A-disk, and the A-disk is considered the "parent" of the B-disk. A disk may have many extensions, but only one level of extension is allowed. If you access an extension A-disk containing no files, the access fails.

How Extensions Are Used

If you have a disk accessed as an extension of another disk, the extension disk is automatically read-only, and you cannot write on it. You might access a disk as its own extension, therefore, to protect the files on it, so that you do not accidentally write on it. For example:

```
access 235 b/b
```

Another use of file extensions is to extend the CMS search order. If you enter a command requesting to read a file, for example:

```
type alpha plan
```

CMS searches your A-disk for the file named ALPHA PLAN and if it does not find it, searches any extensions that your A-disk may have. If you have a file named ALPHA PLAN on your B-disk but have not accessed it as an extension of your A-disk, CMS will not find the file, and you will have to reenter the command:

```
type alpha plan b
```

However, specific commands handle filemode extensions differently. The TYPE command defaults to a filemode of A. Therefore, in the example above, when you did not specify a filemode, CMS would search only the A-disk and all extensions of the A-disk. Refer to the *VM/SP CMS Command Reference* for information on the search order for specific commands.

Note that if you specify a filemode of asterisk (*) or if a particular command defaults to a filemode of asterisk, CMS uses the standard search order (A through Z, without regard to any extensions you have defined). For example, if a certain file was on your W-disk, which you had previously defined as an extension of your A-disk (W/A), CMS would search your A-disk, followed by your B-disk, and any remaining disks alphabetically until it reached the W-disk where the file would be found.

The same applies for those commands which search for a file based on a default filetype, such as the LOAD command which defaults to a filetype of TEXT. Since these commands usually have a default filemode of asterisk, CMS would use the standard search order.

Additionally, for some CMS commands that read and write a file, if you issue the command and the file to be read is on an extension of a read/write disk, the output file is written to the parent read/write disk. The COPYFILE command is a good example of this type of command. If you have a file named FINAL LIST on a B-disk extension of a read/write A-disk, and if you want to make a copy of the file with a different name, type:

```
copyfile final list a final newlist a
```

after the command is completed, the copy is written onto your A-disk. The file on the B-disk remains unchanged.

Accessing and Releasing Read-Only Extensions

When you access a disk as a read-only extension, it remains an extension of the parent disk as long as both disks are still accessed. If either disk is released, the relationship of parent disk/extension is terminated.

If the parent disk is released, the extension remains accessed and you may still read files on it. If you access another disk at the mode letter of the original parent disk, the parent/extension relationship remains in effect.

If you release a read-only extension and access another disk with the same mode letter, it is not an extension of the original parent disk unless you access it as such. For example, if you enter:

```
access 198 c/a
release c
access 199 c
```

the C-disk at virtual address 199 is not an extension of your A-disk.

When to Specify Filemode Letters: Reading Files

When you request CMS to access a file, you have to identify it so that CMS can locate it for you. The commands that expect files of particular filetypes (reserved filetypes) allow you to enter only the filename of the file when you issue the command. When you execute any of these commands or execute a MODULE or EXEC file, CMS searches all of your accessed disks (using the standard search order) to locate the file. Some CMS commands that perform this type of search are:

AMSERV	GLOBAL	MODMAP
ASSEMBLE	LOAD	RUN
DOSLIB	LOADMOD	TXTLIB
EXEC	MACLIB	

Some CMS commands require you to enter the filename and filetype to identify a file. You may specify the filemode letter; if you do not specify the filemode, CMS searches only your A-disk and its extensions when it looks for the file. If you do specify a filemode letter, the disk you specify and its extensions are searched for the file. Some commands you can use this way are:

EDIT	PUNCH	TAPE
FILEDEF	STATE	TYPE
PRINT	SYNONYM	UPDATE

There are some CMS commands that do not search extensions of disks when looking for files. They include:

```
ERASE
FILELIST
LISTFILE
```

You must explicitly enter the filemode if you want to use these commands to list or dump files that are on extensions.

The following commands search every accessed read-only and read-write disk:

```
NAMES
NAMEFIND
```

Using Asterisks and Equal Signs

For some CMS commands, if you specify the filemode of a file as an asterisk, it indicates that you either do not know or do not care what disk the file is on and you want CMS to locate it for you. For example, if you enter:

```
listfile myfile test *
```

the LISTFILE command responds by listing all files on your accessed disks named MYFILE TEST. When you specify an asterisk for the filemode of the COPYFILE, ERASE, or RENAME commands, CMS locates all copies of the specified file. For example:

```
rename temp sort * good sort =
```

renames all files named TEMP SORT to GOOD SORT on all of your accessed read/write disks. An equal sign (=) is valid in output fileids for the RENAME and COPYFILE commands.

For some commands, when you specify an asterisk for the filemode of a file, CMS stops searching as soon as it finds the first copy of the file. For example:

```
type myfile assemble *
```

If there are files named MYFILE ASSEMBLE on your A-disk and C-disk, then only the copy on your A-disk is displayed. The commands that perform this type of search are:

COMPARE	PRINT	STATE
DISK DUMP	PUNCH	SYNONYM
EDIT	RUN	TAPE DUMP
FILEDEF	SORT	TYPE

For the COMPARE, COPYFILE, RENAME, and SORT commands, you must always specify a filemode letter, even if it is specified as an asterisk.

When to Specify Filemode Letters: Writing Files

When you issue a CMS command that writes a file onto one of your virtual disks, and you specify the output filemode, CMS writes the file onto that disk. The commands that require you to specify the output filemode are:

```
COPYFILE
RENAME
SORT
```

The commands that allow you to specify the output filemode, but do not require it, are:

FILEDEF	TAPE LOAD
GENMOD	TAPPDS
READCARD	UPDATE

When you do not specify the filemode on these commands, CMS writes the output files onto your A-disk.

Some CMS commands that create files always write them onto your A-disk. The LOAD and INCLUDE commands write a file named LOAD MAP A5. The LISTFILE command creates a file named CMS EXEC, on your A-disk.

Other commands that do not allow you to specify the filemode, write output files either:

- To the disk from which the input file was read, or
- To your A-disk, if the file was read from a read-only disk.

These commands are:

```
AMSERV
MACLIB
TXTLIB
UPDATE
```

The SORT command also functions this way if you specify the output filemode as an asterisk (*).

In addition, many of the language processors, when creating work files and permanent files, write onto the first read/write disk in your search order, if they cannot write on the source file's disk or its parent.

How Filemode Numbers are Used

Whenever you specify a filemode letter to reference a file, you can also specify a filemode number. Since a filemode number for most of your files is 1, you do not need to specify it. The filemode numbers 0 through 6 are discussed below. Filemode numbers 7 through 9 are reserved for IBM use.

Filemode 0

A filemode number of 0 assigned to a file makes that file private. No other user may access it unless they have read/write access to your disk. Under normal circumstances; if someone links to your disk in read-only mode and requests a list of all the files on your disk, the files with a filemode of 0 are not listed.

The DDR command will allow you to copy the minidisk from one disk to another, and therefore, the filemode 0 files. Use a read share password to protect minidisks with private files when using ACCESS.

Filemode 1

Filemode 1 is used for reading and writing files. It is the default filemode.

Filemode 2

Filemode 2 is essentially the same, for the purposes of reading and writing files, as filemode 1. Usually a filemode of 2 is assigned to files that are shared by users who link to a common disk, like the system disk. Since you can access a disk and specify which files on that disk you want to access, files with a filemode of 2 provide a convenient subset of all files on a disk. For example, if you issue the command:

```
access 489 e/a * * e2
```

you can only read files with a filemode of 2 on the disk at virtual address 489.

Filemode 3

Files with a filemode of 3 are erased after they are read. If you create a file with a filemode of 3 and then request that it be printed, the file is printed, and then erased. You can use this filemode if you write a program or EXEC procedure that creates files that you do not want to maintain copies of on your virtual disks. You can create the file, print it, and not have to worry about erasing it later.

The language processors and some CMS commands create work files and give these work files a filemode of 3.

Note: A filemode of 3 should not be used with EXECs. Depending on what commands are issued within it, an EXEC with a filemode of 3 may be erased before it completes execution.

Filemode 4

Files with a filemode of 4 are in OS simulated data set format. These files are created by OS macros in programs running in CMS. You specify that a file created by a program is to have OS simulated data set format by specifying a filemode of 4 when you issue the FILEDEF command for the output file. If you do not specify a filemode of 4, the output file is created in CMS format.

You can find more details about OS simulated data sets in *VM/SP CMS for System Programming.*

Note: There are no filemode numbers reserved for DOS or VSAM data sets, since CMS does not simulate these file organizations.

Filemode 5

This filemode number is the same, for purposes of reading and writing, as filemode 1. You can assign a filemode of 5 to files that you want to maintain as logical groups, so that you can manipulate them in groups. For example, you can reserve the filemode of 5 for all files that you are retaining for a certain period of time; then, when you want to erase them, you could issue the command:

```
erase * * a5
```

Filemode 6

The filemode number 6 indicates that the "update-in-place" attribute of a CMS file is in effect. This means that the existing records of a file are written back to their previous location on disk rather than in a new slot. This only applies to files located on 512-, 1K-, 2K-, or 4K-byte block formatted minidisks. To take advantage of the "update-in-place" capability, the FSWRITE macro must be used, whether explicitly by the user or implicitly by the system.

CAUTION: It is possible to destroy the integrity of a disk if all of the following conditions are true:

- Updates were made to a filemode number 6 file which altered either the number or length of the records in the file.
- One or more output files remain open on the same disk.
- A system crash or CMS re-IPL occurs.

Note: For a variable format file, "update-in-place" applies only if a record is replaced by a record of the same length.

When To Enter Filemode Numbers

You can assign filemode numbers when you use the following commands:

COPYFILE	You can assign a filemode number when you create a new file with the COPYFILE command.
DLBL, FILEDEF	When you assign file definitions to disk files for programs or CMS command functions, you can specify a filemode number.

GENMOD	You can specify a filemode number with the GENMOD command. To change the filemode number of an existing MODULE file, use the RENAME or COPYFILE commands.
READCARD	You can assign a filemode number when you specify a file identifier with the READCARD command or on a READ control card.
RECEIVE	You can assign a filemode number when receiving a file from your virtual reader.
RENAME	When you specify the fileids on the RENAME command, you can specify the filemode numbers for the input and/or output files. To change only the filemode number of an existing file, you must use the RENAME command. For example: `RENAME test module a1 = = a2` changes the filemode number of the file TEST MODULE A from 1 to 2.
SORT	You can specify filemode numbers for the input and/or output fileids with the SORT command.
XEDIT	You can assign a filemode number when you create a file with the System Product Editor. To change the filemode number of an existing file, use the RENAME or COPYFILE commands, or use the SET FMODE subcommand when you are in the edit environment.

Managing Your CMS Disks

The number of files you can write on a CMS disk depends on both the size of the disk and the size of the files that it contains. You can find out how much space is being used on a disk by using the QUERY DISK command. For example, to see how much space is on your A-disk, you would enter:

```
query disk a
```

The response may be something like this:

```
LABEL   cuu  M  STAT  CYL  TYPE  BLKSIZE  FILES  BLKS USED-(%) BLKS LEFT BLK TOTAL
MYDISK 191  A  R/W    5   3330   1024      171       1221-92         107      1328
```

When a disk is becoming full, you should erase whatever files you no longer need, or dump to tape files that you need to keep but do not need to keep active on disk.

When you are executing a command or program that writes a file to disk, and the disk becomes full in the process, you receive an error message, and you have to try to clear some space on the disk before you can attempt to execute the command or program again. To avoid the delays that such situations cause, you should try to maintain an awareness of the usage of your disks. If you cannot erase any more files from your disks, you should contact installation support personnel about obtaining additional read/write CMS disk space.

CMS File Directories

Each CMS disk has a master file directory that contains entries for each of the CMS files on the disk. When you access a disk, information from the master file directory is brought into virtual storage and written into a user file directory. The user file directory has an entry for each file that you may access. If you have accessed a disk specifying only particular files, then the user file directory contains entries only for those files.

If you have read/write access to a disk, then each time you write the file onto disk the user file directory and master file directory are updated to reflect the current status of the disk. If you have read/write access to a disk and the FSCLOSE macro is issued, the user file directory is updated. When there are no open files on the disk, the master file directory is updated to reflect the current status of the files. If you have read-only access to a disk, then you cannot update the master file directory or user file directory. If you access a read-only disk while another user is writing files onto it, you may need to periodically reissue the ACCESS command for the disk, to obtain a fresh copy of the master file directory.

Warning

You should never attempt to write on a disk at the same time as another user. CMS does not protect a user from loss of data on a disk when multiple users have write access to the disk.

You can use the CP command QUERY LINKS to determine if other users may have links to any disks you may be accessing. Refer to the *VM/SP CP Command Reference* for further information.

The user file directory remains in virtual storage until you issue the RELEASE command specifying the mode letter or virtual address of the disk. If you detach a virtual disk (with the CP DETACH command) without releasing it, CMS does not know that the disk is no longer part of your virtual machine. When you attempt to read or write a file on the disk CMS assumes that the disk is still active (because the user file directory is still in storage) and encounters an error when it tries to read or write the file.

A similar situation occurs if you detach a disk and then add a new disk to your virtual machine using the same virtual address as the disk you detached. For example, if you enter the following sequence of commands:

```
link user1 191 195 rr rpass
access 195 d
detach 195
link user2 193 195 rr rpass
listfile * * d
```

the LISTFILE command produces a list of the files on USER1's 191 disk; if you attempt to read one of these files, you receive an error message. You must issue the ACCESS command to obtain a copy of the master file directory for USER2's 193 disk.

Note: The password cannot be entered on the command line if the password suppression facility was specified when your system was installed.

The entries in the master file directory are sorted alphamerically by filename and filetype, to facilitate the CMS search for particular files. When you are updating disk files, the entries in the user file directory and master file directory tend to become unsorted as files are created, updated, and erased. When you use the RELEASE command to release a read/write disk, the entries are sorted and the master file directory is rewritten.

CMS Command Search Order

When you enter a command in the CMS environment, CMS must locate the command in order to execute it. If you have EXECs in storage or on an accessed disk, or if you have MODULE files on any of your accessed disks, CMS treats them as commands; they are known as user-written commands.

As soon as the command name is found, the search stops and the command is executed. The list below outlines the search order followed each time you specify a command name:

1. Search for an EXEC with the specified command name.

2. Search for a translation or synonym for the specified command name. If one is found, repeat Step 1 to search for an EXEC, using the translation or synonym.

3. Search for a CMS command with the specified command name.

4. Search for a translation or synonym for the specified command name. If one is found, repeat Step 3 to search for a command, using the translation or synonym.

5. Pass the command to CP for execution.

For example, if you specify the command:

```
x sauces cookbook
```

CMS would complete the following search:

1. First, CMS searches for X EXEC. For this example, assume that an X EXEC would not be found.

2. CMS then searches the translation and synonym tables and finds that X is a synonym for XEDIT. Then, CMS repeats Step 1, searching for XEDIT EXEC. Again, assume that an XEDIT EXEC would not be found.

3. Next, CMS searches for a CMS command with the name X. It would not be found.

4. CMS again searches the translation and synonym tables and finds that X is a synonym for XEDIT. Then, CMS repeats Step 3, searching for XEDIT. The XEDIT command would be found and executed. As a result, you would be able to XEDIT the file SAUCES COOKBOOK.

For more detailed information on the CMS command search order, refer to the *VM/SP CMS Command Reference.*

CMS Command Execution Characteristics

Following is an alphabetical list of the CMS commands which require special consideration when invoked from a user program. For example, a program running in the user area cannot call a CMS command which also runs in the user area.

Any commands which are listed in the *VM/SP CMS Command Reference* but are not in this table, are nucleus resident and will not interfere with the execution of a user program.

The "Code" column indicates the execution characteristics of the command.

Code	Meaning
E	indicates that this command is an EXEC. It may execute one or more CMS commands which run in the user or transient areas.
T	indicates that this command executes in the transient area.
U	indicates that this command executes in the user program area. All OS free storage pointers are reset.

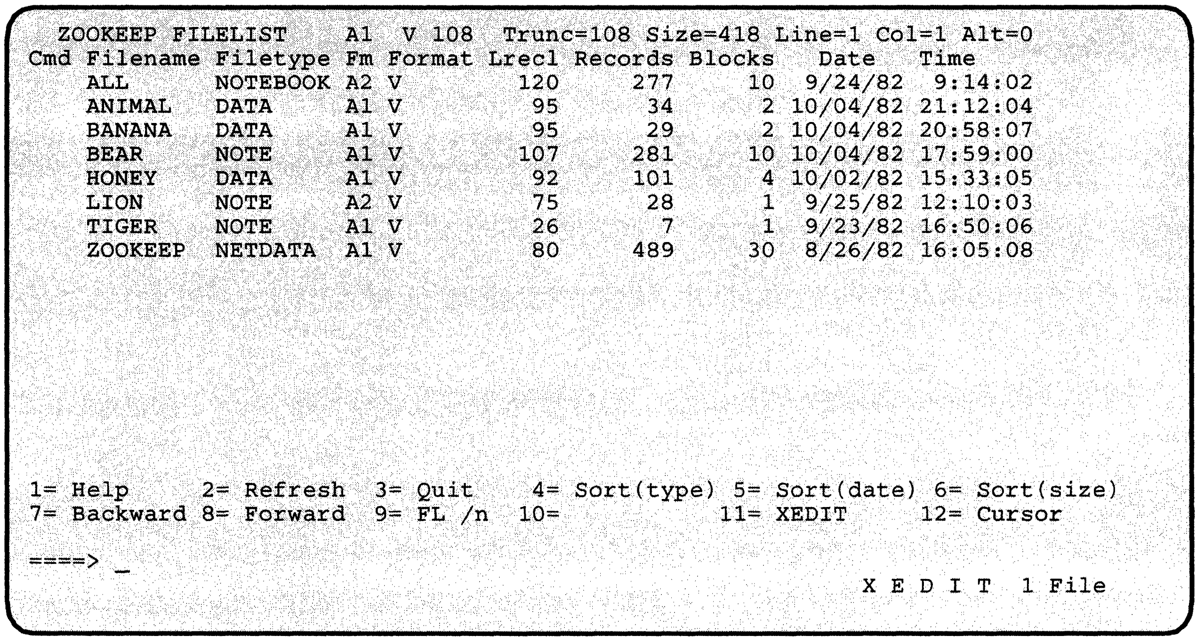
Figure 10. CMS Command Execution Characteristics

Command	Code	Command	Code	Command	Code
AMSERV	U	GENMSG	U	PEEK	E
ASSEMBLE	U	GLOBAL	T	PSERV	U
ASSGN	T	HELPCONV	T	PUNCH	T
CATCHECK	U	HNDINT*	T	RDR	T
CMSBATCH	U	HNDSVC*	T	RDRLIST	E
CMSSERV	E	IOCP	U	READCARD	T
COMPARE	T	LABELDEF	T	RECEIVE	E
CONVERT COMMANDS	E	LANGGEN	E	RESERVE	T
DDR	U	LANGMERG	E	RSERV	U
DEFAULTS	E	LISTDS	U	RUN	E
DISCARD	E	LISTIO	T	SENDFILE	E
DISK	T	LKED	U	SETPRT	T
DOSLIB	U	LOADLIB	U	SORT	U
DOSLKED	U	MACLIB	U	SSERV	U
DOSPLI	E	MACLIST	E	SVCTRACE	T
DSERV	U	MODMAP	T	SYNONYM	T
EDIT	U	MOREHELP	E	TAPE	T
ESERV	E	MOVEFILE	U	TAPEMAC	U
EXECMAP	T	NAMES	E	TAPPDS	U
EXECUPDT	E	NOTE	E	TELL	E
FCOBOL	E	NUCXDROP	T	TXTLIB	U
FILELIST	E	NUCXMAP	T	TYPE	T
FORMAT	U	OPTION	T	UPDATE	U
GENDIRT	T	OSRUN	U		

**Note:* In the list above, HNDINT and HNDSVC are CMS functions, not commands.

Displaying a List of Your CMS Files

Use the FILELIST command to display information about your CMS files residing on accessed disks. In a full screen environment, FILELIST provides you with the same information as the LISTFILE command, but also allows you to edit and issue commands from the list. You can issue XEDIT subcommands to manipulate the list itself. Figure 11 is a sample FILELIST list.



```
  ZOOKEEP FILELIST    A1  V 108  Trunc=108 Size=418 Line=1 Col=1 Alt=0
Cmd Filename Filetype Fm Format Lrecl Records Blocks   Date    Time
    ALL      NOTEBOOK A2 V        120     277     10  9/24/82  9:14:02
    ANIMAL   DATA     A1 V         95      34      2 10/04/82 21:12:04
    BANANA   DATA     A1 V         95      29      2 10/04/82 20:58:07
    BEAR     NOTE     A1 V        107     281     10 10/04/82 17:59:00
    HONEY    DATA     A1 V         92     101      4 10/02/82 15:33:05
    LION     NOTE     A2 V         75      28      1  9/25/82 12:10:03
    TIGER    NOTE     A1 V         26       7      1  9/23/82 16:50:06
    ZOOKEEP  NETDATA  A1 V         80     489     30  8/26/82 16:05:08









1= Help     2= Refresh  3= Quit    4= Sort(type) 5= Sort(date) 6= Sort(size)
7= Backward 8= Forward  9= FL /n  10=           11= XEDIT     12= Cursor

====> _
                                                       X E D I T  1 File
```

Figure 11. Sample FILELIST Screen

Finding Files in Your FILELIST List

If you have many files in your list, the list may take up more than one screen. To find files in your FILELIST list, you can do any of the following:

- Scroll through the list using the PF keys.

Key	Function
PF7	Scrolls backward one full screen.
PF8	Scrolls forward one full screen.

- Rearrange the list using one of the following keys:

Key	Arrangement
PF4	Orders the list by filetype.
PF5	Orders the list by date (newest to oldest). This is how the list is initially arranged.
PF6	Orders the list by size (largest to smallest).

- Use the XEDIT subcommand LOCATE if you know the filename and/or filetype of the file that you are looking for. You enter the LOCATE command at the bottom of the screen and then press the ENTER key. For example:

```
====> locate/banana data/
```

  If BANANA DATA is located, the line containing it becomes the first line on the screen.

- Rearrange the list by entering one of the following synonyms on the command line:

SNAME	Sorts the list alphabetically by filename, filetype, and filemode.
STYPE	Sorts the list alphabetically by filetype, filename, and filemode.
SMODE	Sorts the list by filemode, filename, and filetype.
SRECF	Sorts the list by record format, filename, filetype, and filemode.
SLREC	Sorts the list by logical record length and then by size (greatest to least).
SSIZE	Sorts the list by number of blocks and number of records (greatest to least).
SDATE	Sorts the list by year, month, day, and time (most recent to oldest).

Using FILELIST to List Some of Your Files

FILELIST allows you to obtain various lists of the files on your disks. You can ask for a list of files that have the same filename or filetype or all of the files that begin with a certain letter. The abbreviation for FILELIST is FILEL. Following are various ways that you might use the FILELIST command:

filel	Displays a list of the files on your A-disk.
filel * * b	Displays a list of the files on your accessed B-disk.
filel bear *	Displays a list of the files on your A-disk with a filename of BEAR.
filel * data	Displays a list of files on your A-disk with DATA as the filetype.
filel * * a1	Displays a list of the files with a filemode number 1 on your A-disk.

Erasing Files from FILELIST

Use the DISCARD command to erase from disk a file that is displayed in the list. DISCARD is equivalent to the CMS command ERASE. DISCARD can either be typed in the command area of the line that describes the file you want discarded, or it can be entered from the command line (at the bottom of the screen). DISCARD can only be used while in FILELIST, RDRLIST, MACLIST, and PEEK command environments.

Listing your Files with the LISTFILE command

You can use the LISTFILE command to list information about your CMS disk files. For example, entering:

```
listfile * data
```

lists the files on your A-disk with the filetype of DATA. For example:

```
ANIMAL   DATA     A1
BANANA   DATA     A1
HONEY    DATA     A1
```

If you want more information than just the fileids, you can use one of the information request options for LISTFILE. For example, entering:

```
listfile * data (label
```

returns a list with more than just the fileid. For example:

```
FILENAME FILETYPE FM FORMAT LRECL RECS  BLOCKS  DATE      TIME     LABEL
ANIMAL   DATA     A1 V         95   34       2  10/04/82 21:12:04 ZKP191
BANANA   DATA     A1 V         95   29       2  10/04/82 20:58:07 ZKP191
HONEY    DATA     A1 V         92  101       4  10/02/82 15:33:05 ZKP191
```

As with the FILELIST command, you can vary what you list with the LISTFILE command. Remember you only need to enter L, the minimum truncation for LISTFILE. Following, are various ways that you might use the LISTFILE command:

l	Lists the files on your A-disk.
l * * b	Lists the files on your accessed B-disk.
listf bear *	Lists the files on your A-disk with a filename of BEAR.
l * data	Lists the files on your A-disk with DATA as a filetype.
list * * a1	Lists the files with a filemode number 1 on your A-disk.

Comparing Contents of Files

To compare the contents of two files to see if they are identical, use the COMPARE command. For example:

```
compare labor stat a1 labor stat b1
```

Any records in these files that do not match are displayed at your terminal. The format of the COMPARE command is documented in the *VM/SP CMS Command Reference.*

Copying Files

The COPYFILE command, in its simplest form, copies a file from one virtual disk to another. For example:

```
copyfile linda assemble b pat assemble a
```

Renaming Files

You can change the file identifier of a file with the RENAME command.

```
rename test file a1 good file a1
```

You can use RENAME to modify filemode numbers, for example:

```
rename * module a1 = = a2
```

changes the filemode numbers on all MODULE files on the A-disk that have a mode number of 1 to a mode number of 2. Remember that you cannot use RENAME to move a file from one disk to another. You must use the COPYFILE command to change filemode letters.

Changing the Record Format of a File

Files can either have fixed- or variable-length record formats. You can change the record format of a file with the COPYFILE command and the RECFM option;

```
copyfile data file a (recfm f lrecl 130
```

converts the file DATA FILE A1 to fixed-length 130-character records.

If you want to keep the original file intact, you can specify an output fileid, for example:

```
copyfile data file a fixdata file a (recfm f lrecl 130
```

The file FIXDATA FILE A contains the converted records.

If the records in a file being copied are variable-length, each output record is padded with blanks to the specified record length. If any records are longer than the record length, they are truncated.

When you convert files from fixed-length records to variable-length records, you can specify the COPYFILE TRUNC option to ensure that all trailing blanks are truncated:

```
copyfile data file a (recfm v trunc
```

If you specify the COPYFILE LRECL option and RECFM V, the LRECL option is ignored and the output record length is taken from the longest record in the input file.

When you convert a file from variable-length to fixed-length records, you may also specify a fill character to be used for padding instead of a blank. If you specify:

```
copyfile short recs a (recfm f fill *
```

then each record in the file SHORT RECS is padded with asterisks to the record length. Assuming that SHORT RECS was originally a variable-length file, the record length is taken from the longest existing record. Note that if SHORT RECS is already fixed-length, it is not altered.

Similarly, when you are converting back to variable-length a file that was padded with a character other than a blank, you must specify the FILL option to indicate the pad character, so that character is truncated.

Using Synonyms

By using the SYNONYM and the SET ABBREV commands, you can control what command names, synonyms, or truncations are valid in CMS. For example, you could create a file named MYSYN SYNONYM which contains the following records:

```
PRINT    PRT    1
RELEASE  LETGO  4
FILELIST FL     2
```

The first column specifies an existing CMS command, module, or EXEC name. The second column specifies the alternate name or synonym that you want to use. The third column is a count field that indicates the minimum number of characters of the synonym that can be used to truncate the name. Using this file, after you enter the command:

```
synonym mysyn
```

you can use PRT, LETGO, and FL in place of the corresponding CMS command names. Also, if the ABBREV function is in effect, (it is the default; you can make sure it is in effect by issuing the command SET ABBREV ON), you can truncate any of your synonyms to the minimum number of characters specified in the count field of the record (that is, you could enter P for PRINT and LETG for RELEASE). To invoke your synonym table at the beginning of every terminal session, enter the SYNONYM MYSYN command (or your own synonym table name) into your PROFILE EXEC.

Notes:

1. *An EXEC procedure having a synonym defined for it can be invoked by its synonym if implied EXEC (IMPEX) function is on. However, within an EXEC procedure, only the EXEC filename can be used. A synonym is not recognized within an EXEC because the synonym tables are not searched during EXEC processing.*

2. *You cannot define translations or translation synonyms using the SYNONYM command. Translations must be defined in the Definition Language for Command Syntax (DLCS) file. You can truncate any translation or translation synonym to the minimum number of characters specified in the count field of the record if you issued SET ABBREV ON. See VM/SP CMS for System Programming for more information about DLCS.*

Using Translations

Once you have defined translations and translation synonyms for commands in your DLCS file, you can use the SET TRANSLATE command to control whether or not they are recognized by CMS. The SET ABBREV command controls whether or not the abbreviations of these translations will be recognized.

Note: The translation synonyms defined in a DLCS file are synonyms of command name translations. Do not confuse them with synonyms defined with the SYNONYM command. In the following paragraphs, the term "translation" is used to mean both the command name translations and translation synonyms defined in DLCS. The term "synonym" refers to synonyms defined with the SYNONYM command and abbreviations of system language command names.

When you issue a command in CMS, the command name you use and all of the keywords in it must be in the same language. If you use a translation of the command name, all of the keywords you use with that command will be translated. If you specify a synonym for a command name, the keywords will not be translated. Therefore, you must specify them so that the command will recognize them.

It is possible for the translation of a command or keyword to be the same as the original command or keyword. If the command name you specify is the same as the original command, but you specify keywords for that command in a different language, CMS would determine which language to use upon encountering the first keyword that is different. The subsequent keywords must be in the same language as the first keyword in order to be successfully translated.

There are ways you can code programs and EXECs so that you can choose whether or not to allow translations. CMS only recognizes translations for commands entered from the command line or those invoked with the System Product Interpreter command search function (ADDRESS CMS) or the equivalent search function in EXEC 2 (&PRESUME &SUBCOMMAND CMS).

CMS does not translate your command name or keywords if you SET TRANSLATE OFF. Also, a command will not be translated if it is invoked from another program using the search hierarchy for SVC 202 or using the System Product Interpreter SVC 202 search hierarchy (ADDRESS COMMAND) or the equivalent search function in EXEC 2 (&PRESUME &COMMAND CMS). Refer to *VM/SP CMS for System Programming* for further details on the CMS command search function for translations.

For more details of how CMS uses the translation and synonym tables to find commands, see the section on command resolution in the *VM/SP CMS Command Reference.*

Chapter 4. What You Can Do with CMS Commands

This chapter provides an overview of the kinds of tasks you can perform in CMS and the appropriate commands necessary to perform each of them.

The chapter is organized by task, covering such topics as beginning and ending your terminal session, requesting information, and moving files. Under each task, there is a short discussion of some of the types of things you can do. This is followed by a table containing a list of specific tasks, commands you would use to perform those tasks, and chapters which contain information on the specific commands mentioned.

The commands are not presented in their entirety, nor is a complete selection of commands represented. The purpose of this chapter is to provide you with an understanding of the kinds of commands available to you, so that when you need to perform a particular task using CMS, you may have an idea of whether it can be done, and know what command to reference for details.

For complete lists of the CP and CMS commands available, see Figure 75 on page 302 and Figure 78 on page 313.

Beginning and Ending Your Terminal Session

Your terminal session starts when you logon (with CP LOGON) and ends when you logoff (CP LOGOFF). When you know that you are only going to be away from your terminal for a short while, you can disconnect (CP DISCONN). When you reconnect (with CP LOGON) the status of your virtual machine is the same as you left it.

Task	Command(s)	Description
Begin your terminal session	CP LOGON	Chapter 1
End your terminal session	CP LOGOFF	Chapter 1
	CP DISCONN	Chapter 1

The command formats and usage notes for the commands; DISCONN, LOGOFF, and LOGON, are documented in the *VM/SP CP Command Reference.*

Tailoring Your System

At the start of every terminal session, you can automatically customize your system with the commands invoked by your PROFILE EXEC. Your PROFILE EXEC can include commands to set your program function (PF) keys, access disks, and access your synonym table. For example, you may wish to set up a PF key for the RETRIEVE function to provide you with a quick method of correcting errors without rekeying an entire line of input. Once you have assigned the RETRIEVE function to a PF key, when you press the key, the system will retrieve the line you previously entered.

You may also tailor your system by using full-screen CMS. If you issue SET FULLSCREEN ON at the beginning of your terminal session, or if the command is in your PROFILE EXEC, CMS is in a window and can take advantage of full-screen capabilities. In full-screen CMS, messages and other information can be accessed through windows on your physical screen. When you are using full-screen CMS, you would not need to set a PF key for the RETRIEVE function since the default setting for CMSPF 6 is RETRIEVE.

Your system administrator can also tailor your terminal session for you by making changes to your system profile. These changes would immediately be in effect each time you IPL CMS. For more information about the system profile function, refer to *VM/SP CMS for System Programming.*

You can also write your own EXEC to execute a series of commands. You could then invoke the EXEC from your PROFILE EXEC.

Another way you could customize your system is by specifying the language you wish to use for entering commands and receiving messages.

When you are communicating with others on your computer, use the NAMES command to assign nicknames. The nicknames can be used with the SENDFILE and TELL commands because both commands reference your NAMES file. By setting up nicknames, you can tailor your system to allow you to send messages and notes much more quickly.

Action	Command(s)	Description
Set your program function (PF) keys	CP SET PFnn	Chapter 1
Assign a PF key to retrieve previously entered lines	CP SET PFnn RETRIEVE	Chapter 1
Keep information about others with whom you communicate	NAMES	Chapter 8

Action	Command(s)	Description
Specify defaults for the commands: FILELIST, NOTE, PEEK, RDRLIST, RECEIVE, and SENDFILE	DEFAULTS	Chapter 8
Assign synonyms for system and your own commands	SYNONYM	Chapter 3
Tailor your System at the start of every session via your PROFILE EXEC	System Product Editor and the System Product Interpreter	Chapter 13
Write your own command that executes several commands or programs	System Product Editor and the System Product Interpreter	Chapter 12
Make EXECs storage resident	EXECLOAD	Chapter 13
Access EXECs in a DCSS	SET INSTSEG	Chapter 14
Specify a language for entering CMS commands and receiving system messages	SET LANGUAGE	Chapter 1
Invoke full-screen CMS	SET FULLSCREEN ON	Chapter 9
Display information in windows	POP WINDOW DROP WINDOW	Chapter 9
Customize full-screen CMS	DEFINE WINDOW DEFINE VSCREEN POSITION WINDOW SIZE WINDOW MAXIMIZE WINDOW RESTORE WINDOW	Chapter 15

The CP SET command is documented in the *VM/SP CP Command Reference*. The command formats and usage notes for the following CMS commands are documented in the *VM/SP CMS Command Reference*:

```
DEFAULTS
DEFINE VSCREEN
DEFINE WINDOW
DROP WINDOW
EXECLOAD
MAXIMIZE WINDOW
NAMES
POP WINDOW
POSITION WINDOW
RESTORE WINDOW
SET FULLSCREEN ON
SET LANGUAGE
SIZE WINDOW
SYNONYM
XEDIT
```

Requesting Information

You can use CP and CMS commands to inquire about your terminal, virtual machine, disks, data files, members of a CMS library, and other users.

Inquiring about:	Command(s)	Description
Terminal characteristics	CP QUERY SCREEN CP QUERY SET	Chapter 1 Chapter 1
Languages that are currently active in your virtual machine	CP QUERY CPLANG QUERY LANGUAGE	Chapter 1
Languages that are available for your virtual machine	QUERY LANGLIST	Chapter 1
Files on your disk	FILELIST LISTFILE	Chapter 3 Chapter 3
Files in your reader	RDRLIST CP QUERY RDR ALL	Chapter 8 Chapter 1,8
Your spool files	CP QUERY FILES	Chapter 8
Your virtual disks	QUERY DISK QUERY SEARCH	Chapter 3 Chapter 3
Your virtual machine	IDENTIFY	Chapter 14
Your print files	CP QUERY PRINTER	Chapter 7
Your reader, printer, and punch	CP QUERY UR	Chapter 7
Other users	CP QUERY userid	Chapter 8
Storage Resident EXECs	EXECMAP	Chapter 13

The command format and usage notes for the CP QUERY command are found in the *VM/SP CP Command Reference.* Command formats and usage notes for the following CMS commands can be found in the *VM/SP CMS Command Reference.*

```
EXECMAP
FILELIST
IDENTIFY
LISTFILE
QUERY
RDRLIST
```

Communicating with Other Computer Users

You can use CP and CMS commands to send files, notes and messages to one or more users on your system or a system that is attached to yours via Remote Spooling Communications Subsystem (RSCS) network. The NOTE, SENDFILE, and TELL commands reference your "userid NAMES" file. The names file, created with the NAMES command, contains a collection of information about other computer users with whom you communicate.

Action	Commands	Description
Creating Names file	NAMES	Chapter 8
Sending files	SENDFILE	Chapter 8
Sending messages	TELL	Chapter 8
Sending notes	NOTE and SENDFILE	Chapter 8

The following commands are CMS commands:

```
NAMES
NOTE
SENDFILE
TELL
```

Their command formats and usage notes are documented in the *VM/SP CMS Command Reference.*

Controlling Terminal Output

VM/SP allows you to control your terminal output. You can refuse messages with the CP SET MSG command. If you only want to see the short form of the CMS ready message, you set this with the CMS SET RDYMSG command.

The CP SCREEN command allows you to select colors and extended highlighting on certain 3279 models.

If your program is directing output to your terminal, you can halt the terminal display with the HT Immediate command, and later resume terminal display with the RT Immediate command.

Action	Command(s)	Description
Alter any extended color or highlighting definitions	CP SCREEN	Chapter 1
Control whether or not you receive messages	CP SET	Chapter 1
Indicate the type of CMS ready message that you want	SET RDYMSG	Chapter 1
Suppress terminal output	HT Immediate command	Chapter 2
Resume terminal output that was previously suppressed via HT	RT Immediate command	Chapter 2

The command formats for the CP commands:

```
SCREEN
SET
```

are documented in the *VM/SP CP Command Reference.* The command formats for the CMS commands:

```
HT (Immediate command)
RT (Immediate command)
SET
```

are documented in the *VM/SP CMS Command Reference.*

Sharing Virtual Disks

VM/SP allows you to share virtual disks on either a permanent or temporary basis. You can add another user's disk to your configuration with the CP LINK command. When you no longer need a disk that you have linked to or have temporarily accessed, you can release it with the CMS RELEASE command. When you no longer need a disk in your virtual machine configuration, you can disconnect it with the CP DETACH command.

Action	Command(s)	Description
Establish a link to a disk	CP LINK	Chapter 1
	ACCESS	Chapter 1
Release a disk	RELEASE	Chapter 1
	CP DETACH	Chapter 1

The formats and usage notes for the following CP commands:

```
DETACH
LINK
```

are documented in the *VM/SP CP Command Reference.*

The formats and usage notes for the following CMS commands:

```
ACCESS
RELEASE
```

are documented in the *VM/SP CMS Command Reference.*

What You Can Do to the Files in Your Virtual Reader

CMS and CP commands allow you to look at, get rid of, keep, load onto disk, and reorder the files in your virtual reader.

Action	Command(s)	Description
Display a list of your reader files	RDRLIST	Chapter 8
Look at a file	PEEK	Chapter 8
Load the file onto your disk	RECEIVE	Chapter 8
Purge a file	DISCARD (when in PEEK or RDRLIST)	Chapter 8
	CP PURGE	Chapter 8
Transfer a file to (or from) the reader queue of another user	CP TRANSFER	Chapter 8
Alter the external attributes of a file	CP CHANGE	Chapter 7
Change the order of the files	CP ORDER	Chapter 7

The CMS commands:

```
DISCARD
PEEK
RECEIVE
RDRLIST
```

are documented in the *VM/SP CMS Command Reference.*

The CP commands:

```
CHANGE
ORDER
PURGE
TRANSFER
```

are documented in the *VM/SP CP Command Reference.*

Receiving or Loading Files onto Your Disk

Files that are in your reader or on a tape can be loaded onto your disk.

Retrieving files from ...	Command(s)	Description
Your virtual reader	RECEIVE	Chapter 8
A tape	TAPE LOAD	Chapter 7
	FILEDEF and MOVEFILE	Chapter 7

Command formats and usage notes for the following CMS Commands are documented in the *VM/SP CMS Command Reference.*

```
FILEDEF
MOVEFILE
RECEIVE
TAPE LOAD
```

Erasing Files from Your Virtual Disk

When you no longer need a file you can erase or discard it from your disk. You can use the ERASE command when you want to erase all the files with a particular filemode letter and number or the files with the same filename or filetype. You can use the DISCARD command when in FILELIST to erase files. You can also use DISCARD from PEEK and RDRLIST environments. This is discussed under "What You Can Do to the Files in Your Virtual Reader." The FORMAT command erases *all* files on a particular disk.

Action	Command(s)	Description
Erase specific files	ERASE	Chapter 1
Erase files from FILELIST menu	DISCARD	Chapter 3
Erase all files on a particular disk	FORMAT	Chapter 1

The *VM/SP CMS Command Reference* contains information on the following CMS commands that you can use to erase files:

```
DISCARD
ERASE
FORMAT
```

Creating and Modifying Files

The System Product Editor, invoked with the XEDIT command, allows you to interactively make changes, additions, or deletions to your CMS files. The UPDATE command and the XEDIT command with the UPDATE option provide a way for you to modify a source program without affecting the original.

Action	Command(s)	Description
Edit a file	XEDIT	Chapter 6

The command formats and usage notes for the UPDATE and XEDIT commands are documented in the *VM/SP CMS Command Reference.*

Moving Files

CMS commands allow you to move a file or copies of a file from one place to another; from one virtual disk to another, to or from a tape to a disk, or from your disk to the virtual reader of another user. Some commands, such as the TAPE command, move a copy of the file to another location. Other commands, such as the CP TRANSFER command, move (not copy) files.

Action	Command(s)	Description
Move files from your virtual reader to the reader of another virtual machine	CP TRANSFER	Chapter 8
Dump contents of a virtual disk onto tape, restore such files to disk	DDR	Chapter 7
Move files from tapes to disk, disk to tape	TAPE	Chapter 7
Move files from your disk to the reader of another (or your own) virtual machine	SENDFILE	Chapter 8
Move files from your virtual reader onto your read/write disk	RECEIVE	Chapter 8

The CP TRANSFER command is documented in the *VM/SP CP Command Reference.*

The formats and usage notes for the following CMS commands:

```
DDR
RECEIVE
SENDFILE
TAPE
```

are documented in the *VM/SP CMS Command Reference.*

Chapter 5. Using the HELP Facility

The VM/SP HELP facility can assist you in your work by allowing you to conveniently display information about commands you may wish to use or to display explanatory information on system messages.

This chapter will give you a general understanding of the structure and function of the VM/SP HELP facility. It will get you started using HELP and give you some examples to practice with. If you need specific information on the format and syntax of HELP and related commands, refer to the *VM/SP CMS Command Reference.*

The HELP facility has the following components:

Component	**Description**
CP	Control Program commands
CMS	Conversational Monitor System commands
CMSSET	CMS SET command options
CMSQUERY	CMS QUERY command options
CPSET	CP SET command options
CPQUERY	CP QUERY command options
SET	XEDIT SET command options
QUERY	XEDIT QUERY command options
IPCS	Interactive Problem Control System commands
DEBUG	DEBUG subcommands
EDIT	EDIT subcommands
EXEC	EXEC statements
EXEC 2	EXEC 2 statements
XEDIT	XEDIT subcommands
REXX	System Product Interpreter statements
SQLDS	SQL/Data System Program Product (5748-XXJ) (only if you have this installed on your system.)
SRPI	Server-Requester Programming Interface subcommands
TSAF	Transparent Services Access Facility

The HELP facility is designed for use on 3270-type terminals in display mode. It can also be used on line-oriented terminals. All the information about commands and messages is contained in files called HELP files. In display mode, HELP uses the System Product Editor to display these files. In line-mode, the HELP facility types the HELP file to your screen one line at a time.

When you access the HELP facility, HELP will EXECLOAD the HELPXED XEDIT macro if it has not been loaded. Loading this file into storage improves the performance of the HELP command. If you occasionally use HELP, you may want to EXECDROP the HELPXED XEDIT macro to release the storage. Refer to the *VM/SP CMS Command Reference* for further information.

HELP files will normally appear on your screen in mixed case. However, in some installations, lowercase characters may be reserved for displaying special alphabets. In this case, HELP files will be displayed all in uppercase. For further details, refer to the *VM/SP Installation Guide.*

As you work through the examples in this chapter and use HELP, you may need to refer to the "DETAIL HELP" section in Chapter 16, "Tailoring the HELP Facility," in order to understand the format boxes in the command HELP files.

You may notice when you perform certain tasks within the HELP facility that your PF key listing at the bottom of the screen is covered by a message. If this occurs, simply press ENTER to refresh the screen.

Getting HELP on Commands

To understand the HELP facility, it is best to look first at its structure and second at how it functions.

There are two different types of HELP files, information files and selection files. Information files contain information about commands and messages. Selection files display menus that allow the user to select the appropriate information file. This section explains the use of information files. Refer to the section entitled "Menus" on page 99 for an explanation of selection files.

The HELP facility is designed to provide various ways of getting information for a command, depending on your level of expertise and the amount of detail you require for a particular task. Commands can contain three layers of information: BRIEF, DETAIL, and RELATED. Each layer displays a unique level of HELP.

You may display any of the three available layers by specifying the corresponding layering options: BRIEF, DETAIL, or RELATED. You may specify only one layering option at a time. However, once you have requested one layer of HELP on a specified command, you may toggle (switch) between the other layers available for that command. See "Toggling" on page 101 for a further explanation of how to toggle between layers of HELP.

BRIEF is the default option, meaning that if you do not specify a layering option, the BRIEF layer of HELP will be displayed if it exists. If BRIEF HELP is not available for a certain command, DETAIL HELP will be displayed. The following sections provide a further explanation of the three layers of command HELP.

Note: The examples used within this chapter include the component name (CP, CMS, REXX, etc.). However, if you are requesting HELP for a command in CP or CMS, you do not need to specify the component name. For example, to request HELP on the CMS command SENDFILE, you could simply specify HELP SENDFILE. For components other than CP or CMS (REXX, EXEC, TSAF, and so forth), you must specify the component name in order for HELP to locate the proper command information. For example, in order to get help on the REXX instruction, TRACE, you would need to specify HELP REXX TRACE.

BRIEF HELP

BRIEF is the first of three layers of HELP and is available for many commands. BRIEF HELP displays a short description of the requested command, its basic syntax (command without options), an example, and, if applicable, a message telling the user that either more or related information is available.

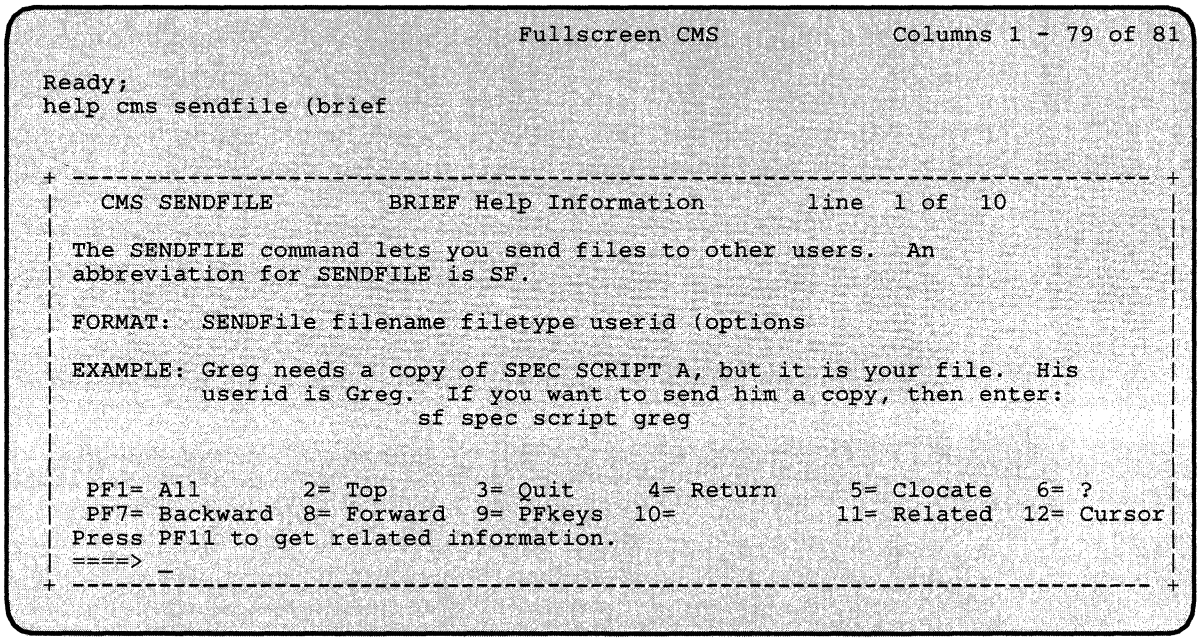
If you are in full-screen CMS and request BRIEF HELP, your screen will show the HELP command you issued and just below it, will display the BRIEF HELP information in a window which will be displayed on your screen. If you are not in full-screen CMS, your entire screen will display the BRIEF HELP information.

The following example shows how your screen would look if you requested BRIEF HELP for the SENDFILE command in full-screen CMS.

If you are not currently in full-screen CMS, enter the command SET FULLSCREEN ON. Then, type:

```
help cms sendfile (brief
```

The following screen will be displayed:



```
                              Fullscreen CMS            Columns 1 - 79 of 81

Ready;
help cms sendfile (brief


+ ----------------------------------------------------------------------- +
|   CMS SENDFILE        BRIEF Help Information       line  1 of  10         |
|                                                                           |
| The SENDFILE command lets you send files to other users.  An              |
| abbreviation for SENDFILE is SF.                                          |
|                                                                           |
| FORMAT:  SENDFile filename filetype userid (options                       |
|                                                                           |
| EXAMPLE: Greg needs a copy of SPEC SCRIPT A, but it is your file.  His    |
|          userid is Greg.  If you want to send him a copy, then enter:     |
|                       sf spec script greg                                 |
|                                                                           |
|                                                                           |
|  PF1= All       2= Top      3= Quit     4= Return     5= Clocate   6= ?   |
|  PF7= Backward  8= Forward  9= PFkeys  10=           11= Related  12= Cursor|
| Press PF11 to get related information.                                    |
| ====> _                                                                   |
+ ----------------------------------------------------------------------- +
```

Figure 12. Sample of BRIEF HELP for the SENDFILE Command

Since the remainder of the examples in this chapter will not be done in full-screen CMS, press PF3 to quit the HELP screen. Then enter the command SET FULLSCREEN OFF before proceeding with the next example.

DETAIL HELP

The DETAIL layer of HELP presents a complete description of the command, the command format, an explanation of its parameters and options, usage notes, and error information. DETAIL HELP provides information similar to the information you would find listed in the *VM/SP CMS Command Reference.*

Subsetting Options

The DETAIL layer of HELP has seven subsetting options: DESCRIPT, FORMAT, PARMS, OPTIONS, NOTES, ERRORS, and ALL. By specifying subsetting options, the user can display one or more particular sections of the DETAIL HELP. ALL is the default option, meaning that the entire DETAIL HELP will be displayed. It is possible to change the default option, but if you do so, you will need to specify ALL as the subsetting option to display the entire DETAIL layer. Refer to the DEFAULTS command in the *VM/SP CMS Command Reference* for more details. The *VM/SP CMS Command Reference* also contains a complete description of the subsetting options.

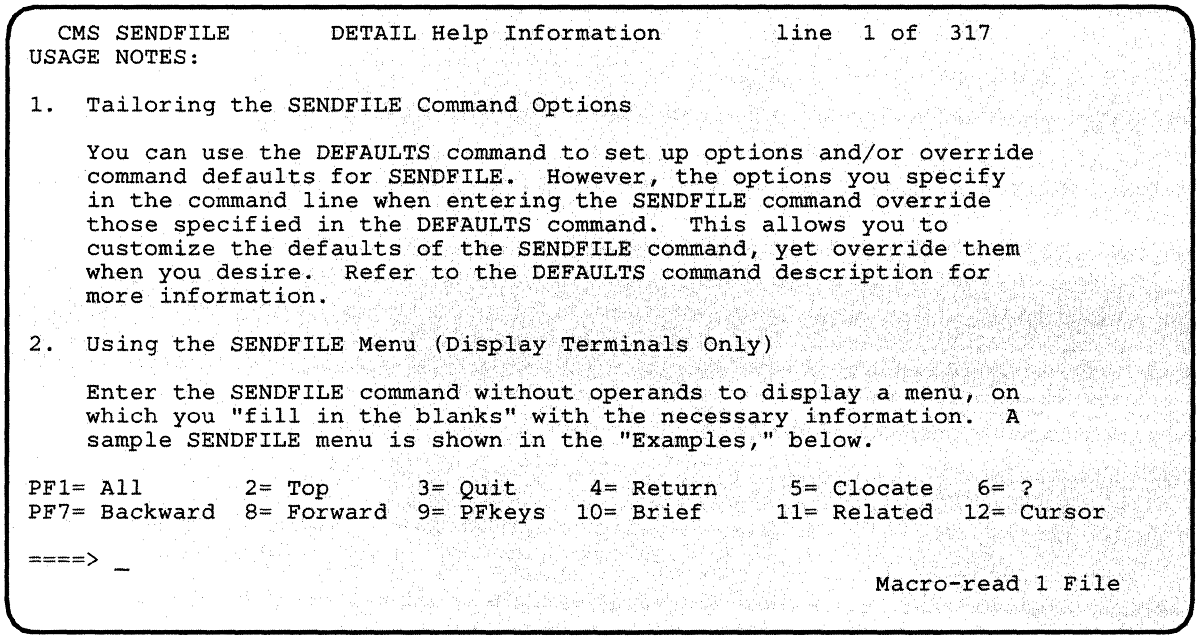
For example, to display the entire DETAIL layer of the SENDFILE command in the CMS environment, you would type:

```
help cms sendfile (detail
```

Let's display just the usage notes in that same file. Type:

```
help cms sendfile (notes
```

The following screen will be displayed:



```
  CMS SENDFILE       DETAIL Help Information        line  1 of  317
USAGE NOTES:

1.  Tailoring the SENDFILE Command Options

    You can use the DEFAULTS command to set up options and/or override
    command defaults for SENDFILE.  However, the options you specify
    in the command line when entering the SENDFILE command override
    those specified in the DEFAULTS command.  This allows you to
    customize the defaults of the SENDFILE command, yet override them
    when you desire.  Refer to the DEFAULTS command description for
    more information.

2.  Using the SENDFILE Menu (Display Terminals Only)

    Enter the SENDFILE command without operands to display a menu, on
    which you "fill in the blanks" with the necessary information.  A
    sample SENDFILE menu is shown in the "Examples," below.

PF1= All       2= Top      3= Quit     4= Return     5= Clocate   6= ?
PF7= Backward  8= Forward  9= PFkeys  10= Brief     11= Related  12= Cursor

====> _
                                                          Macro-read 1 File
```

Figure 13. Sample of DETAIL HELP for the SENDFILE Command

RELATED HELP

The RELATED layer of HELP is a multi-purpose layer. RELATED HELP is designed to make the user aware of commands that are similar to the presently displayed HELP file, making it easier for the user to decide which command to use. For example, if you want to remove a file from your readerlist and, after reading HELP ERASE, you realize that ERASE is not the correct command, the RELATED layer of the ERASE command will let you easily access the HELP file for the correct command, DISCARD.

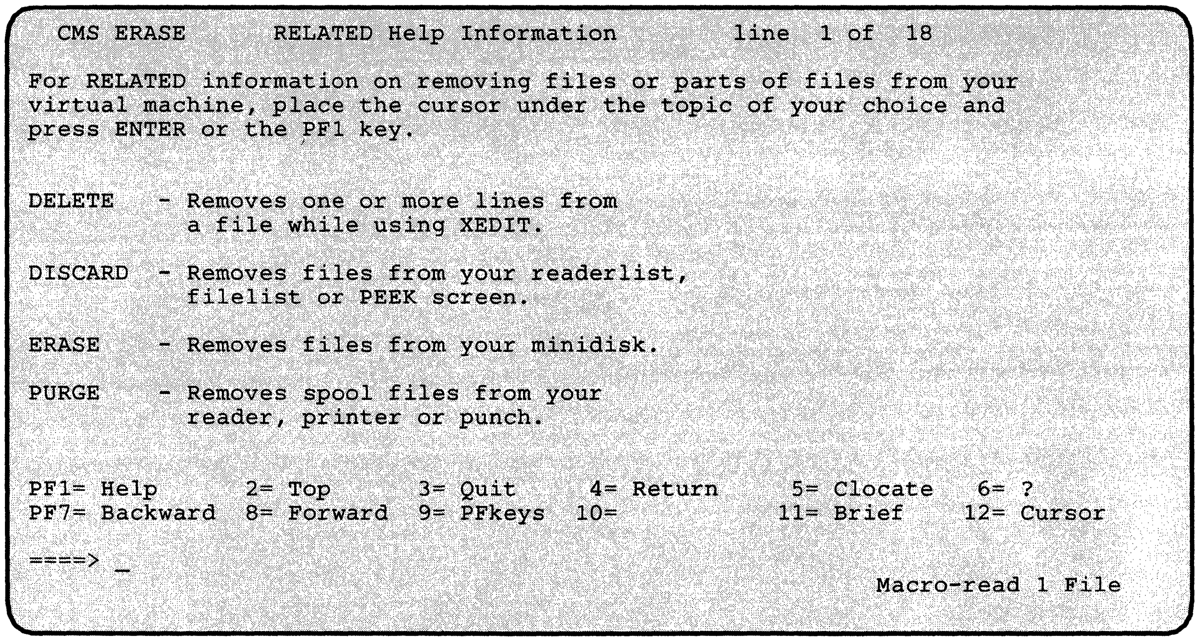
When you request RELATED HELP on the SET or QUERY commands, the screen will list and briefly describe all the SET and QUERY operands available for the system component. You could access HELP information on any of the displayed operands directly from these menu screens by positioning the cursor on a particular operand and pressing ENTER.

For information on how to create your own RELATED HELP files refer to Chapter 16, "Tailoring the HELP Facility."

For example, to display the RELATED layer of the ERASE command in the CMS environment, type:

```
help cms erase (related
```

The following screen will be displayed:



```
  CMS ERASE      RELATED Help Information        line  1 of  18

For RELATED information on removing files or parts of files from your
virtual machine, place the cursor under the topic of your choice and
press ENTER or the PF1 key.

DELETE    - Removes one or more lines from
            a file while using XEDIT.

DISCARD   - Removes files from your readerlist,
            filelist or PEEK screen.

ERASE     - Removes files from your minidisk.

PURGE     - Removes spool files from your
            reader, printer or punch.

PF1= Help      2= Top     3= Quit     4= Return     5= Clocate    6= ?
PF7= Backward  8= Forward 9= PFkeys  10=           11= Brief     12= Cursor

====> _
                                                      Macro-read 1 File
```

Figure 14. Sample of RELATED HELP for the ERASE Command

Other Options

There are five other options which affect the display of HELP: SCREEN/NOSCREEN, TYPE/NOTYPE, and EXTEND. Briefly, these other options control the display of files and error messages and the search order of commands. A complete description of these options can be found in the HELP command section of the *VM/SP CMS Command Reference.*

Getting HELP on SET and QUERY

The HELP facility provides a special feature which allows you to quickly access HELP on specific SET or QUERY options in CP, CMS, or XEDIT. If you know the component name (CP, CMS, or XEDIT) and the name of the option, you can immediately get to the specific HELP file you need.

Reserved filetypes in the HELP facility allow you to access specific HELP files. In order to access HELP information for any CMS SET option, you would type HELP, followed by the filetype, CMSSET, and the option name. For example, to get HELP on the CMS command SET RDYMSG, you would type the following:

```
help cmsset rdymsg
```

This command would immediately access the specific HELP file for the CMS SET RDYMSG command.

To request HELP for a CMS QUERY option, you would type HELP, followed by the filetype CMSQUERY, and the option name.

The same is true for CP SET and QUERY options. To get HELP on a CP SET option, you would type HELP, followed by CPSET, and the option name; for HELP on a CP QUERY option, you would type HELP, CPQUERY, and the option name.

To request HELP for XEDIT SET and QUERY options, you can simply type HELP, followed by SET or QUERY, and the name of the option. For example, to get HELP on the XEDIT command SET APL, you would type the following:

```
help set apl
```

Even though there is also a CMS command SET APL, the HELP facility would correctly provide you with HELP information on the XEDIT SET APL command. You would need to type:

```
help cmsset apl
```

to request information on the CMS SET APL command.

For more information on HELP for SET and QUERY options, refer to the description of the HELP command in the *VM/SP CMS Command Reference*.

Facts About Command HELP

The following information will help you to more efficiently use command HELP:

- Command HELP is designed to allow you to easily move between screens of HELP information. While viewing the command HELP for one command, you can get help for another command by specifying "HELP" followed by the name of the new command. For instance, if you were viewing the HELP file for the CMS PRINT command, and decided you wanted help on the CMS COPYFILE command, you could enter the following on the command line of your current screen:

  ```
  help cms copyfile
  ```

  This would bring you directly to the HELP information for the COPYFILE command.

- BRIEF, DETAIL, and RELATED are conflicting HELP options. You can specify only one of these options in the command string. If you specify more than one option at a time, the last option entered will be effective. For example, if you typed the following on the command line:

  ```
  help cms erase (brief related
  ```

  RELATED HELP for the ERASE command will be displayed.

  If you typed the following on the command line:

  ```
  help cms erase (descript notes brief
  ```

  BRIEF HELP for the ERASE command would be displayed. If you then entered MOREHELP on the command line or pressed PF10, you would see the DESC and NOTES sections for the ERASE command. Use of the PF keys to move between HELP screens is explained in detail in the section entitled "Using the PA2 and PF Keys" on page 101.

Getting HELP on Messages

Sometimes, when you perform a CMS task, the system will respond with a message. In order to find out why the message was produced and perform any necessary corrective action, you can refer to Message HELP.

The HELP files for messages display the message text, an explanation of why the message was issued, the system action, and a user action. For a complete description of all the possible ways to request HELP for messages, refer to the *VM/SP CMS Command Reference.*

For example, to display the HELP file for the CMS message DMSHLP002E, type the following on the command line:

```
help DMS002E
```

or

```
help DMSHLP002E
```

Menus

In VM/SP HELP, menus serve as selection files, that is, HELP panels that assist you in reaching the HELP files you need. If you are uncertain of the name of a component or command you might like to use, you can simply choose from a list of likely possibilities.

You can select an entry from the menu by positioning the cursor in front of or under any part of the entry and then pressing ENTER or the PF1 key. After the HELP file is displayed, you may return to the menu by pressing PF3.

To position the cursor at the entry you want, you can do any one of the following:

- Use the key marked --->|, which functions as a tab key, causing the cursor to move to the first character of the next entry.
- Use another cursor-movement key.
- Type on the command line the desired entry or a string that appears in the task description and press PF5.

When the cursor is positioned at the desired entry, press ENTER or the PF1 key to display the HELP file for that entry.

There are two types of menus, task menus and component menus. An asterisk (*) preceding an entry in a displayed menu indicates that the entry is the name of a component menu file. A colon (:) indicates that the entry is the name of a task menu.

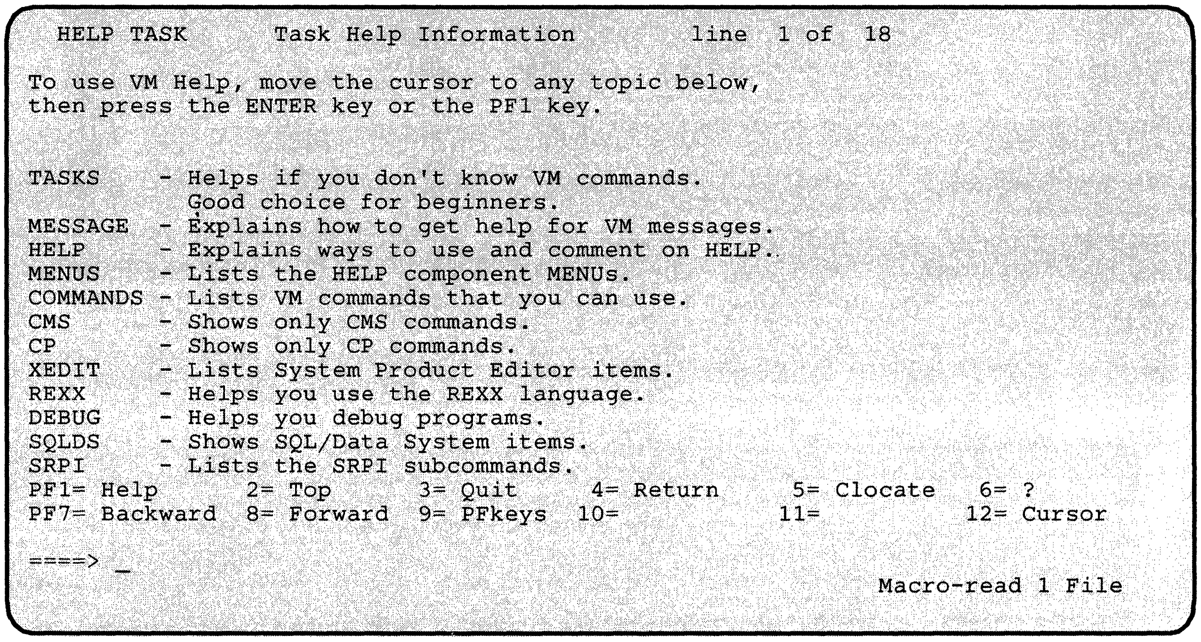
TASK Menus in HELP

TASK menus are especially useful for users new to the VM system because they describe an action that the user may want to perform and then guide the user to the appropriate HELP file.

You can see a list of tasks and components available to you by typing:

```
help task
```

The following TASK menu will be displayed:



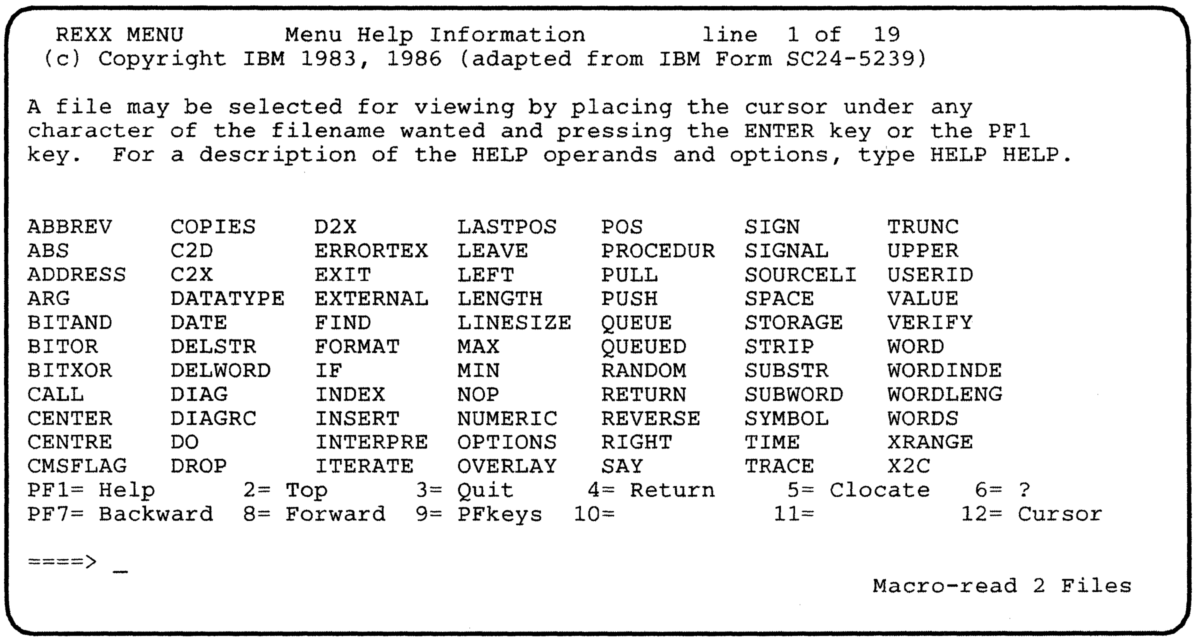
```
  HELP TASK      Task Help Information        line  1 of  18

To use VM Help, move the cursor to any topic below,
then press the ENTER key or the PF1 key.


TASKS     - Helps if you don't know VM commands.
            Good choice for beginners.
MESSAGE   - Explains how to get help for VM messages.
HELP      - Explains ways to use and comment on HELP.
MENUS     - Lists the HELP component MENUs.
COMMANDS  - Lists VM commands that you can use.
CMS       - Shows only CMS commands.
CP        - Shows only CP commands.
XEDIT     - Lists System Product Editor items.
REXX      - Helps you use the REXX language.
DEBUG     - Helps you debug programs.
SQLDS     - Shows SQL/Data System items.
SRPI      - Lists the SRPI subcommands.
PF1= Help       2= Top       3= Quit     4= Return      5= Clocate    6= ?
PF7= Backward  8= Forward  9= PFkeys  10=           11=           12= Cursor

====> _
                                                          Macro-read 1 File
```

Figure 15. Sample HELP TASK Menu

Component Menus

Component MENUs list the names of all the command HELP files available for a specific component.

If you followed the example shown in the section above, your screen will now be displaying the HELP TASK menu. To display all the command HELP files available for REXX, position your cursor anywhere under the word REXX, and press ENTER. The following component MENU will be displayed:



```
  REXX MENU       Menu Help Information       line  1 of  19
 (c) Copyright IBM 1983, 1986 (adapted from IBM Form SC24-5239)

A file may be selected for viewing by placing the cursor under any
character of the filename wanted and pressing the ENTER key or the PF1
key.  For a description of the HELP operands and options, type HELP HELP.


ABBREV     COPIES     D2X        LASTPOS    POS        SIGN       TRUNC
ABS        C2D        ERRORTEX   LEAVE      PROCEDUR   SIGNAL     UPPER
ADDRESS    C2X        EXIT       LEFT       PULL       SOURCELI   USERID
ARG        DATATYPE   EXTERNAL   LENGTH     PUSH       SPACE      VALUE
BITAND     DATE       FIND       LINESIZE   QUEUE      STORAGE    VERIFY
BITOR      DELSTR     FORMAT     MAX        QUEUED     STRIP      WORD
BITXOR     DELWORD    IF         MIN        RANDOM     SUBSTR     WORDINDE
CALL       DIAG       INDEX      NOP        RETURN     SUBWORD    WORDLENG
CENTER     DIAGRC     INSERT     NUMERIC    REVERSE    SYMBOL     WORDS
CENTRE     DO         INTERPRE   OPTIONS    RIGHT      TIME       XRANGE
CMSFLAG    DROP       ITERATE    OVERLAY    SAY        TRACE      X2C
PF1= Help       2= Top       3= Quit      4= Return      5= Clocate    6= ?
PF7= Backward  8= Forward   9= PFkeys   10=            11=           12= Cursor

====> _
                                                          Macro-read 2 Files
```

Figure 16. Sample Component MENU for REXX

If you type HELP MENUS, your screen will display a listing of all the available menus.

Using the PA2 and PF Keys

The PA2 key (or its equivalent) and PF keys have special meanings when in the HELP facility. This section provides details on the settings for each key.

As you use the PF keys, you will note that the last PF key you press will appear highlighted on your screen.

Toggling

A toggle key is a key that allows you to move back and forth between various HELP displays. Toggle keys for HELP are PF1, PF10, and PF11. These keys allow you to toggle (switch) between the BRIEF, DETAIL, ALL, and RELATED HELP sections.

The type of information you will get when you press each of these keys depends on the type of HELP information available for a particular command and also depends on what information is being displayed on your screen at the time when you press the PF key. The PF key settings shown at the bottom of your HELP screen will change to reflect the kinds of

HELP available to you. If a certain type of HELP is not available for a particular command, the corresponding PF key setting at the bottom of your HELP screen will be blank.

The PF1 key toggles between the ALL and BRIEF layers of HELP. PF10 toggles between the DETAIL and BRIEF layers of HELP. PF11 toggles between RELATED and BRIEF. For example, if you are viewing the BRIEF layer of HELP, your PF keys would be set as follows:

```
PF1  = ALL
PF10 = MOREHELP
PF11 = RELATED
```

This means that if you were to press PF10, you would then receive MOREHELP, which would provide you with the DETAIL layer of HELP information. Now, while you were viewing DETAIL HELP, your PF keys would have changed to the following settings:

```
PF1  = ALL
PF10 = BRIEF
PF11 = RELATED
```

If you pressed PF10, you would return to BRIEF HELP, and your PF key settings would be as they were in the first part of this example.

The following table lists the settings for PF1, PF10, and PF11 when all HELP layers are available for the displayed file:

Screen Display	PF Key Settings

BRIEF	PF1 = ALL
	PF10 = MOREHELP
	PF11 = RELATED

DETAIL	PF1 = ALL
	PF10 = BRIEF
	PF11 = RELATED

ALL	PF1 = BRIEF
	PF10 = blank
	PF11 = RELATED

RELATED	PF1 = HELP
	PF10 = MOREHELP
	PF11 = BRIEF

Figure 17. Toggling Between Layers of HELP

Following is a listing of the values for PA2 and the PF keys. On a terminal equipped with 24 PF keys, PF keys 13 to 24 are assigned the same values as PF keys 1 to 12.

Key	Meaning	Usage
PA2	Print	is used to print a hard copy of currently displayed HELP information. Remember that after quitting HELP, you must issue CP SP PRT CLOSE in order to print the file.
PF1	Help	accesses HELP files from a menu or a RELATED section after the cursor is positioned at the desired entry.
	All	displays the HELP file as if the ALL option was specified.
	Brief	displays BRIEF information when viewing the ALL option of a specified HELP file. *Note:* If PF1 appears blank on a HELP screen, this means that either ALL or BRIEF HELP is not available for a particular command.
PF2	Top	moves the display to the beginning of the HELP file.
PF3	Quit	exits from the currently displayed HELP file.
PF4	Return	exits from the HELP facility. PF4 quits all HELP files currently in storage. For example, if you call a menu, then call a HELP file from that menu, PF4 quits both the file and the menu and returns control to the originating environment.
PF5	Clocate	is the XEDIT subcommand CLOCATE. On the command line, enter the string you are looking for. Then press PF5 to tell HELP to locate the next occurrence of the string. Repeated pressing of the PF5 key locates additional occurrences of the string. HELP highlights the line located.
PF6	?	displays the last user command issued from the command line.
PF7	Backward	scrolls the display backward one screen.
PF8	Forward	scrolls the display forward one screen.
PF9	PF Key	displays a file containing an explanation of PF key meanings for displayed files.

Figure 18 (Part 1 of 2). PA and PF Keys in the HELP Facility

Key	Meaning	Usage
PF10	Morehelp	displays the HELP information from a command file as if the DETAIL option was specified.
	Brief	displays BRIEF information. *Note:* If PF10 appears blank, this means that either DETAIL HELP or BRIEF HELP is not available for a particular command. In this instance, PF1 would be set to ALL and provide you with all the HELP available for the command.
PF11	Related	displays the HELP information from a command file as if the RELATED option was specified.
	Brief	displays BRIEF information. *Note:* If PF11 appears blank, this means that either RELATED HELP or BRIEF HELP is not available for a particular command.
PF12	Cursor	moves the cursor to the command line or to its previous location on the screen.

Figure 18 (Part 2 of 2). PA and PF Keys in the HELP Facility

Using the MOREHELP Command

If you are viewing HELP facility files on a line-mode terminal, or if you cannot use a PF key to obtain DETAIL or RELATED information, you may find the MOREHELP command useful.

MOREHELP provides you with either additional or related information about the last valid HELP command you issued. For more information on the MOREHELP command, refer to the *VM/SP CMS Command Reference.*

The System Product Editor

If you are using HELP in display mode on a 3270-type terminal, the HELP facility uses the System Product Editor to display HELP files. Many of the features of the XEDIT subcommands are available for use on the displayed files. Two of the available features are:

Clocate	Locates a specified character string in the file or uses PF5 to search the file. PF5 positions the cursor under the target string. CLOCATE remembers the string that was last used, even if you have performed other functions. You can press PF5 to use that last string again. To change the string being searched, enter a new string on the command line and press PF5 to search for the new string.
Scrolling	Moving the display up or down.

See the *VM/SP System Product Editor Command and Macro Reference* for a complete explanation of these features.

Not all XEDIT subcommands are available for use on the displayed HELP files. The excluded subcommands are listed below:

```
FILE           REPLACE
INPUT          SET
MACRO          POWERINP
READ
```

These subcommands are excluded to prevent unnecessary copying of HELP files or to avoid any inadvertent changes to the HELP files. If you use these subcommands while viewing a HELP file, they will be ignored, and you will receive an error message. While these subcommands will not work on files displayed by the HELP facility, you can use them when you use the XEDIT subcommand to edit the files.

Note: When you issue a command from the command line, the system does not search for XEDIT subcommand synonyms and XEDIT macros such as the SPLTJOIN macro. This means that when a macro such as SPLTJOIN is invoked from within the HELP facility, the message 'No such subcommand SPLTJOIN' will be issued.

Printing HELP Screens

When you display HELP files, you can get a printed copy of the screen by pressing PA2 while the screen is displayed. To receive a hard copy:

1. Press the PA2 key.

2. Exit from HELP.

3. Type CP SP PRT CLOSE, and press the ENTER key.

Note: If you want to print the entire HELP file for a certain command, you will need to access the HELP disk and print the file. See your system administrator for information on accessing and printing HELP files.

Working with Your HELP Files

Now that you are familiar with using HELP files, you may wish to learn how to customize your VM/SP HELP files. Refer to Chapter 16, “Tailoring the HELP Facility” for details.

Part 2: Working with VM/SP

Once you are familiar with the basics of the VM/SP system, you are ready to begin using VM/SP to perform routine tasks such as editing files, using input and output devices, communicating with other users, and using the CMS batch facility. Part 2 will provide you with the detailed information you need to perform these tasks.

Chapter 6: Editing Your Files contains some of the basic information you need to create and write a disk file directly from your terminal, or to correct or modify an existing CMS file.

Chapter 7: Using Real Printers, Punches, Readers, and Tapes discusses how to use tapes and punched cards in CMS, and how to use your virtual printer and punch to get real output.

Chapter 8: Communicating with Other Computer Users discusses the ways in which you can send information to other users and can receive information from them.

Chapter 9: Looking at VM/SP Through Windows describes how to invoke full-screen CMS and use windowing functions to view information in windows on your physical screen.

Chapter 10: Using the CMS Batch Facility describes how to prepare and send job streams to a CMS batch virtual machine. The CMS batch facility is a CMS feature that allows you to send jobs to another machine for execution.

Chapter 6. Editing Your Files

To edit a file means to make changes, additions, or deletions to a CMS file that is on a disk, and to make these changes interactively: you instruct the editor to make a change, the editor does it, and then you request another change. You can edit a file that does not exist; when you do so, you create the file online, and can modify it as you enter it.

To file a file means to write a file you are editing back onto a disk, incorporating any changes you made during the editing session. When you issue the FILE subcommand to write a file, you are no longer in edit environment, but are returned to the CMS environment. You can, however, write a file to disk and then continue editing it, by using the SAVE subcommand.

An editing session is the period of time during which a file is in your virtual storage area, from the moment you issue the XEDIT command or the EDIT command until you let the editor know that you are finished working on the file, by entering FILE or QUIT.

When you have set full-screen CMS on and you are using XEDIT, CMS output, as well as messages and other information, will be displayed in windows which will automatically appear on your XEDIT screen. If a window appears, move the cursor to any corner of the window border and type a "F" which will scroll the window forward and cause it to disappear from your screen. You can also type a "C" to clear the window.

This feature of XEDIT in full-screen CMS is particularly useful because it allows you to view your messages without being removed from the XEDIT environment. For more information on using full-screen CMS, refer to Chapter 9, "Looking at VM/SP through Windows."

The System Product Editor

The editor provides the following capabilities:

1. Full screen support for IBM 3270 Display Terminals is available including:

   - The ability to display multiple views of the same file or of different files
   - Automatic "wrapping" of lines that are wider than a screen line

- The ability to enter selected (prefix) subcommands directly on the displayed lines
- The ability to define the screen format according to individual preferences.

2. Extended string search facilities are provided for improved text processing.
3. A variety of macros that use the EXEC 2 or System Product Interpreter are offered.
4. An enhanced set of functions to handle program development is available, including automatic update generation.
5. The ability to import and export data between files is provided.
6. The ability to edit and manipulate files that contain Double-Byte Character Set (DBCS) strings.

   Many languages have more characters than can be displayed using one-byte codes (KANJI, for example). A Double-Byte Character Set (DBCS) is used to represent those characters. The double-byte characters may appear in a sentence with characters from other languages that are displayed in one-byte codes. Files containing double-byte characters are handled differently than files that only contain one-byte characters. Special considerations for editing files that contain double-byte characters are described in the *VM/SP System Product Editor Command and Macro Reference.*

For complete information about the System Product Editor, see the *VM/SP System Product Editor User's Guide* and the *VM/SP System Product Editor Command and Macro Reference.*

The XEDIT Command

When you issue the XEDIT command you must specify the filename and filetype of the file you want to edit. For example:

```
xedit newfile script
```

Writing a File Onto Disk

A file you create and the modifications that you make to it during an edit session are not automatically written to a disk file. To save the results, you can do the following:

- Periodically issue the subcommand:

  ```
  save
  ```

  to write onto disk the contents of the file as it exists when you issue the subcommand. Periodically issuing this XEDIT subcommand protects your data against a system failure; you can be sure that changes you make are not lost.

- At the beginning of the edit session, issue the SET AUTOSAVE subcommand, with a number:

  ```
  set autosave 10
  ```

  Then, for every tenth change or addition to the file, the editor issues an automatic save request, which writes the file onto disk.

- To terminate the editing session and write the file onto disk, issue the subcommand:

  ```
  file
  ```

  The file disappears from your screen, but the editor saved it on your disk. You can return to the edit environment by issuing the XEDIT command, specifying a different file or the same file.

  The editor decides which disk to write the file onto according to the following hierarchy:

  - If you specify a filemode on the FILE or SAVE subcommand line, the file is written onto the specified disk.
  - If the current filemode of the file is the mode of a read/write disk, the file is written onto that disk. (If you have not specified a filemode letter, it defaults to your A-disk.)
  - If the filemode is the mode of a read-only extension of a read/write disk, the file is written onto the read/write parent disk.
  - If the filemode is the mode of a read-only disk that is not an extension of a read/write disk, the editor cannot write the file and issues an error message.

If you are editing a file and decide that you do not wish to save the changes, you can use the subcommand:

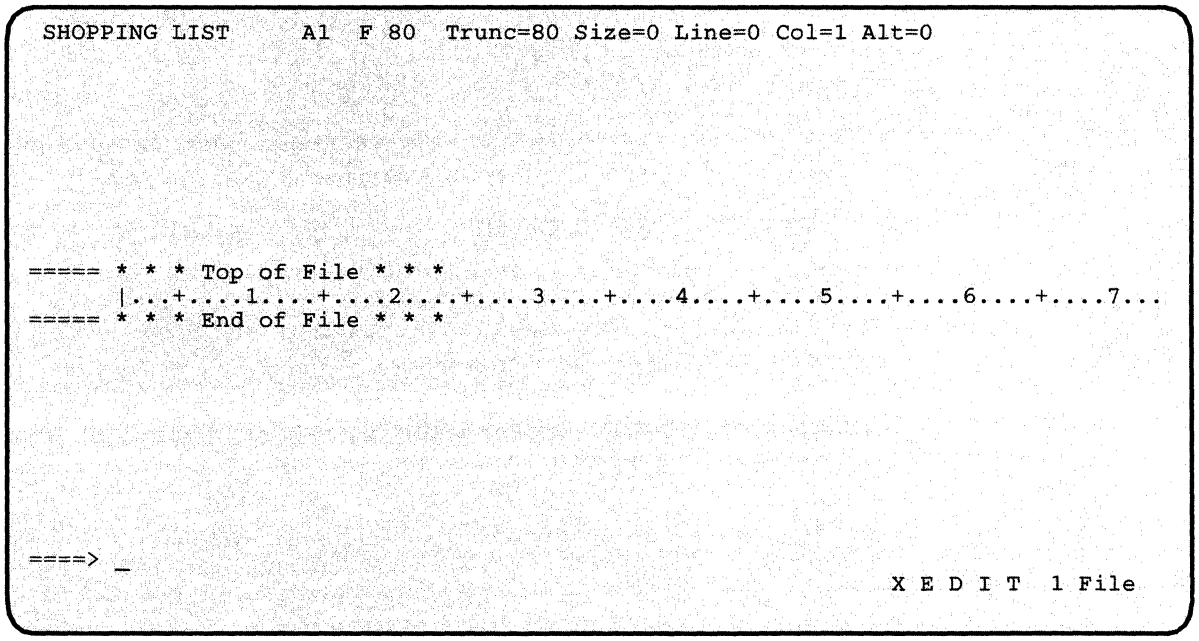
```
quit
```

No changes that you made since you last used the SAVE subcommand (or the editor last issued an automatic save for you) are retained. If you have just begun an edit session, and have made no changes at all to a file, and for some reason you do not want to edit it at all (for example, you misspelled the name, or want to change a CMS setting before editing the file), you can use the QUIT subcommand instead of the FILE subcommand to terminate the edit session and return to CMS.

A file must have at least one line of data in order to be written. To create a new file called SHOPPING LIST, enter:

```
xedit shopping list
```

The XEDIT command invokes the System Product Editor, so what you will see looks like Figure 19.



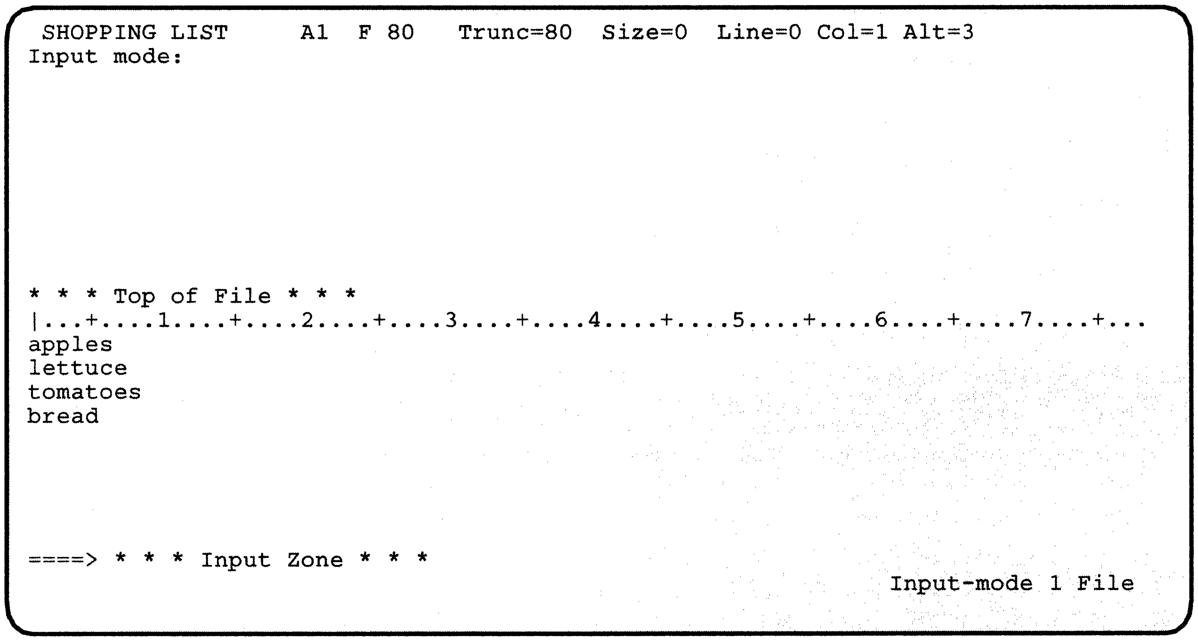
```
 SHOPPING LIST      A1  F 80  Trunc=80 Size=0 Line=0 Col=1 Alt=0








===== * * * Top of File * * *
      |...+....1....+....2....+....3....+....4....+....5....+....6....+....7...
===== * * * End of File * * *









====> _
                                                          X E D I T  1 File
```

Figure 19. Sample XEDIT Screen

On the command line (next to the arrow) type INPUT and press the ENTER key. The file is placed in input mode. The cursor is placed automatically on the first line in the input zone, where you can enter your data. You are writing input lines that are eventually going to be written onto your A-disk.

Enter the following data:

```
apples
lettuce
tomatoes
bread
```



```
 SHOPPING LIST      A1   F 80   Trunc=80  Size=0  Line=0 Col=1 Alt=3
Input mode:









* * * Top of File * * *
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+...
apples
lettuce
tomatoes
bread





====> * * * Input Zone * * *
                                                              Input-mode 1 File
```

Figure 20. Sample XEDIT Screen in INPUT Mode

Now, press the ENTER key and the screen moves up so that you can enter more data. When you are finished entering data, press the ENTER key again to return to edit mode.

To keep this file in permanent storage, you type FILE on the command line and press the ENTER key. You should see a message that looks something like this:

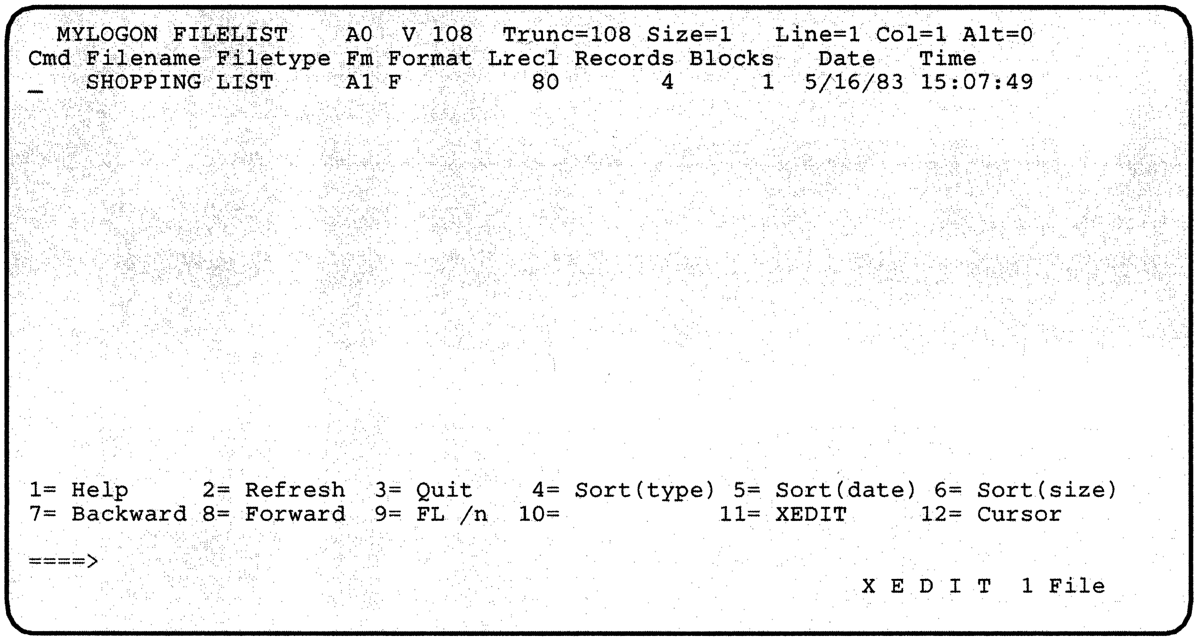
```
Ready;
```

Even though the file has disappeared from your screen, the editor has saved it on your disk.

Let's check and see if the file was really saved. We'll use the FILELIST command to list the files on your A-disk with a filename of shopping. Enter:

```
filelist shopping
```

The display may look like Figure 21.



```
  MYLOGON FILELIST    A0  V 108  Trunc=108 Size=1   Line=1 Col=1 Alt=0
Cmd Filename Filetype Fm Format Lrecl Records Blocks   Date    Time
_   SHOPPING LIST     A1 F         80       4      1  5/16/83 15:07:49









1= Help      2= Refresh  3= Quit     4= Sort(type) 5= Sort(date) 6= Sort(size)
7= Backward 8= Forward  9= FL /n  10=            11= XEDIT     12= Cursor

====>
                                                        X E D I T  1 File
```

Figure 21. Sample FILELIST Screen for a Particular Filetype

Press the PF3 key to leave the filelist screen.

What To Do When You Run Out of Space

There are two situations that may prevent you from continuing an edit session or from writing a file onto disk:

- The editor may run out of virtual storage.
- Your disk space may become full.

You should be aware of these situations, know how to avoid them, and how to recover from them, should they occur.

When You Run Out of Virtual Storage

When you issue the XEDIT command to edit a file, the editor copies the file into virtual storage. If it is a large file, or you have made many additions to it, the editor may run out of storage space. If it does, it issues the message:

```
No storage available
```

When this happens, you cannot make any changes or additions to the file unless you first delete some lines. If you attempt to add a line, the editor issues the message:

```
No storage available to insert lines
```

You should use the FILE subcommand to write the file onto disk. If you want to continue editing, you should see that the editor has more storage space to work with. To do this, you can find out how large your virtual machine is and then increase its size. To find out the size, issue the CP QUERY command:

```
query virtual storage
```

If the response is:

```
STORAGE =  1024K
```

you might want to redefine your storage to 2M. Use the CP command DEFINE, as follows:

```
define storage 2m
```

This command resets your virtual machine, and you must issue the CP IPL command to reload the CMS system before you can continue editing.

If a file is very large, the editor may not have enough space to allow you to edit it using the XEDIT command. The message:

```
DMSXIN132S File fn ft fm too large
```

indicates that you must obtain more storage space before you can edit the file. If this is the case, or if you are editing large files, you should redefine your storage before beginning the terminal session with the CP DEFINE command. For example to increase the size of your virtual storage to 4M, enter:

```
define storage 4m
```

This resets your virtual machine and you must IPL CMS again. If you issue the CP DEFINE command and receive the message:

```
Storage exceeds allowed maximum
```

you should see your installation support personnel about having the directory entry for your userid updated so that you have a large storage size to begin with.

Splitting CMS Files Into Smaller Files

If the file you are editing is too large, and the data it contains does not have to be in one file, you can split the file into smaller files, so that it is easier to work with. Two of the methods you can use to do this are described below.

Use the COPYFILE Command: You can use the COPYFILE command to copy portions of a file into separate files, and then delete the copied lines from the original file. For example, if you have a file named TEST FILE that has 1000 records, and you want to split it into four files, you could enter:

```
copyfile test file a test1 file a (from   1 for 250
copyfile test file a test2 file a (from 251 for 250
copyfile test file a test3 file a (from 501 for 250
copyfile test file a test4 file a (from 751 for 250
```

When these COPYFILE commands are complete, you have four files containing the information from the original TEST FILE, which you can erase:

```
erase test file
```

Use the XEDIT command: If you use the editor to create smaller files, you can edit them as you copy them, that is, if you have other changes that you want to make to the data. To copy files with the editor, you use the GET subcommand. Using the file TEST FILE as an example, you might enter:

```
xedit test1 file
get test file a 1 250
.
.
.
file
xedit test2 file
get test file a 251 250
.
.
.
```

Again, you could erase the original TEST FILE when you are through with your edit session.

When Your Disk Is Full

When you enter a FILE or SAVE subcommand or when an automatic save request is issued, the editor writes a copy of the file you are editing onto disk, and names it XEDTEMP CMSUT1. If this causes the disk to become full, the editor erases the work file, and you receive the message:

```
Disk is full; set new filemode or clear some disk space
```

The original file (as last written onto disk) remains unchanged. You can use the FILELIST command to list the files on the disk, then the DISCARD command to erase the unwanted files.

If you cannot erase any of the files on the disk, there are several alternate recovery paths you can take:

1. If you have another read/write disk accessed, you can use the FMODE subcommand to change the filemode of the file, so that when you file it, it is written to the other disk. If you have a read/write disk that is not accessed, you can access it. After filing the file on the second disk, erase the original copy, and then use the COPYFILE command to transfer the file back to its original disk.

2. If you do not have any other read/write disk in your virtual machine, you may be able to transfer some of your files to another user, using either the SENDFILE, PUNCH or DISK command. When the files have been read onto the other user's disk, you can erase them from your disk. Then, return to edit mode and issue the FILE subcommand.

3. In CMS subset, erase the original disk file (if it existed), then return to edit mode and file the copy that you are editing. You should not use this method unless absolutely necessary, since any unexpected problems may result in the loss of both the disk file and the copy.

After you use the FILE subcommand to write the file onto disk, you should continue erasing any files you no longer need.

Using the Editor in Line-Mode

The editor's display mode is the most common format of operation on a 3270. There are, however, instances when it is not possible or not desirable to use the editor in display mode. For these instances, you should use the line-mode of operation, which is the equivalent to using a typewriter terminal. When you use line-mode, each XEDIT subcommand you enter, and the response (if you have verification on), is displayed, a line at a time, on the screen in the output display area.

You need only be concerned with using line-mode if you are connected to VM/SP by a remote 3270 line, or if you are editing a file from within an EXEC and you want to control the screen display. Although it is possible to use the editor in line-mode on a local 3270, it is rarely necessary for normal editing purposes.

Editing on a Remote 3270

When you invoke the editor from a remote 3270, you are placed in line-mode by the editor. The advantage of using the 3270 in line-mode (particularly on a remote editor) is that the editor can respond more quickly to display requests. When you use display mode, the editor has to write out the entire output display area when you move the current line pointer; in line-mode, it has only to write a single line.

If you want to use display mode, you enter the XEDIT subcommand:

```
set terminal display
```

The editor begins operating in display mode, and you can use the special editing functions available in display mode.

However, when you are using a remote 3270 in display mode, and you enter the INPUT subcommand to begin entering input lines, the screen is cleared, and your input lines are displayed as if you were in line-mode, beginning at the top of the screen. When you enter a null line to return to edit mode, the editor returns to a full screen display.

You can resume editing in line-mode by using the subcommand:

```
set terminal typewriter
```

Editing From an EXEC File

If you invoke the editor from an EXEC, but you do not want the screen cleared when the editor gets control, you can specify the NOCLEAR option on the XEDIT command line:

```
xedit test file (noclear
```

Entering the command:

```
xedit test file (noscreen
```

places the 3270 in line-mode, so that the lines already on the screen are not erased. The 3270 remains in line-mode for the remainder of the edit session, and you cannot use the SET TERMINAL subcommand to place it in display mode.

Chapter 7. Using Real Printers, Punches, Readers, and Tapes

CMS Unit Record Device Support

CMS supports one virtual reader at address 00C, one virtual punch at address 00D, and one virtual printer at address 00E. When you invoke a CMS command or execute a program that uses one of these unit record devices, the device must be attached at the virtual address indicated.

Using the CP Spooling System

Any output that you direct to your virtual printer or punch, or any input you receive from your reader, is controlled by the spooling facilities of the control program (CP). Each output unit is known to CP as a spool file, and is queued for processing with the spool files of other users on the system. Ultimately, a spooled printer file or a spooled punch file may be released to a real printer or card punch for printing or punching.

The final disposition of a unit record spool file depends on the spooling characteristics of your virtual unit record devices, which you can alter with the CP command SPOOL. To find out the current characteristics of your unit record devices you can issue the command:

```
query ur
```

Figure 22 is an example of the response you will receive from issuing this command.

```
RDR 00C CL A  NOCONT   HOLD   EOF       READY
PUN 00D CL A  NOCONT NOHOLD COPY 001    READY FORM STANDARD
    00D TO  CMSGDE   DIST 2G47-706  DEST OFF
PRT 00E CL A  NOCONT NOHOLD COPY 001    READY FORM STANDARD
    00E TO  CMSGDE   DIST 2G47-706  FLASHC 000  DEST OFF
    00E FLASH       CHAR       MDFY       FCB
```

Figure 22. CP QUERY Unit Record Response

You can use the SPOOL command to change spool file characteristics. When you use the SPOOL command to control a virtual unit record device, you do not change the status of spool files that already exist, but rather set the characteristics for subsequent output. For information on modifying existing spool files, see "Altering Spool Files" below.

Spool File Characteristics

Note: When you issue a SPOOL command for a unit record device, you can refer to it by its virtual address, as well as by its generic device type (for example, CP SPOOL E HOLD).

CLASS (CL):

Spool files, in the CP spool file queue, are grouped according to class, and all files of a particular class may be processed together, or directed to the same real output device. The default values for your virtual machine are set in your VM/SP directory entry, and are probably the standard classes for your installation.

You may need, however, to change the class of a device if you want a particular type of output, or some special handling for a spool file. For example, if you are printing an output file that requires special forms, and your installation expects that output to be spooled class Y, issue the command:

```
spool printer class y
```

All subsequent printed output directed to your printer at virtual address 00E (all CMS output) is processed as class Y.

HOLD:

If you place a HOLD on your printer or punch, any files that you print or punch are not released to the control program's spooling queue until you specifically alter the hold status. By placing your output spool files in a hold status, you can select which files you print or punch, and you can purge duplicate or unwanted files. To place printer and punch output files in a hold status issue the commands:

```
spool printer hold
spool punch hold
```

When you have placed a hold status on printer or punch files and you produce an output file for one of these devices, CP sends you a message to remind you that you have placed the file in a hold:

```
PRT FILE xxxx FOR userid COPY xx HOLD
```

If, however, you have issued the command:

```
set msg off
```

then you do not receive the message.

When you place a reader file in a hold status, then the file remains in the reader until you remove the hold status and read it, or you purge it.

COPY:

If you want multiple copies of a spool file, you should use the COPY operand of the SPOOL command:

```
spool printer copy 10
```

If you enter this command, then all subsequent printer files that you produce are each printed 10 times, until you change the COPY attribute of your printer.

FOR:

You can spool printed or punched output so that it will be distributed to another userid by using the FOR operand of the SPOOL command. For example, if you enter:

```
spool printer for charlie
```

then, all subsequent printer files that you produce have, on the output separator page, the userid CHARLIE and the distribution code for that user. The spool file is then under the control of that user, and you cannot alter it further.

CONT, NOCONT:

You can print or punch separate spool files with the NOCONT option of the CP SPOOL command. You can also combine them into one continuous spool file if you use the CONT operand of the CP SPOOL command. For example, if you issue the following sequence of commands:

```
spool punch cont to brown
punch asm1 assemble
punch asm2 assemble
punch asm3 assemble
spool punch nocont
close punch
```

Then, the three files ASM1 ASSEMBLE, ASM2 ASSEMBLE, and ASM3 ASSEMBLE, are punched to user BROWN as a single spool file. When user BROWN reads this file onto a disk, however, CMS creates separate disk files.

You may send multiple files by continuous spooling (using CP SPOOL PUNCH CONT) or by a series of DISK DUMP commands, but these methods are discouraged. As a sender, you are encouraged to do the following:

1. Always use SENDFILE, which resets any continuous spooling options in effect.

2. Do not spool the punch continuous.

Similarly, if the punch is spooled continuous and PUNCH is used to send multiple files, the file is read in as one file with ":READ" cards imbedded. In this case, although no files are overlaid, the recipient must divide the file into individual files. This problem can also be avoided by using SENDFILE or by not spooling the punch continuous.

TO:

When you spool your printer or punch to another userid, all output from that device is transferred to the virtual reader of the userid you specify. When you are punching a CMS disk file, as in the example above, you should use the TO operand of the SPOOL command to specify the destination of the punch file.

You can also use this operand to place output in your own virtual reader by using the * operand:

```
spool printer to *
```

After you enter this command, subsequent printed output is placed in your virtual reader. You might use this technique as an alternative way of preventing a printer file from printing, or, if you choose to read the file onto disk from your reader, of creating a disk file from printer output.

Similarly, if you are creating punched output in a program and you want to examine the output during testing, you could enter:

```
spool punch to *
```

so that you do not punch any real cards or transfer a virtual punch file to another user.

Altering Spool Files

After you have requested that VM/SP print or punch a file, or after you have received a file in your virtual reader and before the file is actually printed, punched, or read, you can alter some of its characteristics, change its destination, or delete it altogether.

Every spool file in the VM/SP system has a unique four-digit number from 1 to 9900 assigned to it, called a spoolid. You can use the spoolid of a file to identify it when you want to do something to it. You can also change a group of files, by specifying that all files of a particular class be altered in some way, or you can manipulate all of your spool files for a certain device at the same time.

The CP commands that allow you to manipulate spool files are CHANGE, ORDER, PURGE, and TRANSFER. In addition, you can use the CP

QUERY command to list the status and characteristics of spool files associated with your userid.

When you use any of these commands to reference spool files of a particular device, you have the choice of referring to the files by class or by spoolid. You can also specify ALL. For example, if you enter the command:

```
query printer all
```

you might see the display:

```
ORIGINID FILE CLASS RECORDS CPY HOLD DATE  TIME     NAME     TYPE    DIST
CMSUG    0142 K PRT 000178  002 USER 04/17 07:58:48 SCHED    SCRIPT  BIN706
CMSUG    0180 1 PRT 002021  001 NONE 04/17 08:02:26 TESTFILE SCRIPT  BIN706
```

Until any of these files are processed, or in the case of files in the hold status, until they are released, you can change the spool file name and spool file type (this information appears on the first page or first card of output), the distribution code, the number of copies, the class, or the hold status, using the CP CHANGE command. For example:

```
change printer all nohold
```

changes all printer files that are in a hold status to a nohold status. The CP CHANGE command can also change the spooling class, distribution code, and so on.

If you decide that you do not want to print a particular printer file, you can delete it with the CP PURGE command:

```
purge printer 7615
```

After you have punched a file to some other user, you cannot change its characteristics or delete it unless you restore it to your own virtual reader. You can do this with the TRANSFER command:

```
transfer all from usera
```

This command returns to your virtual reader all punch files that you spooled to USERA's virtual reader.

You can determine, for your reader or printer files, in what order they should be read or printed. If you issue the command:

```
order printer 8195 6547
```

Then, the file with spoolid of 8195 is printed before the file with a spoolid of 6547.

The CP spooling system is very flexible, and can be a useful tool, if you understand and use it properly. The *VM/SP CP Command Reference* contains complete format and operand descriptions for the CP commands you can use to modify spool files.

Using Your Card Punch and Card Reader in CMS

The CMS RECEIVE command reads files from your virtual reader and places them on disk. Files can be placed in the reader in one of three ways:

- Reading real punched cards into the system card reader. A CP ID card tells the CP spooling system which virtual reader is to receive the file.
- Receiving a file sent to your virtual machine from another user. The SENDFILE or NOTE commands may be used to send a file to the virtual reader of another virtual machine.
- Using the SENDFILE command to send a file to your own reader.

Using Real Cards

If you have a deck of punched cards that you want read into your virtual machine reader, you should punch, preceding the deck, a CP ID card. If your userid is MCGUIRE, then the id card would be:

```
ID MCGUIRE
```

Then, to read this file onto your CMS A-disk, you can enter the command:

```
receive = prog6 assemble
```

If a file named PROG6 ASSEMBLE already exists, you must specify the replace option on the RECEIVE command:

```
receive = prog6 assemble (replace
```

If you are reading many files into the real system card reader, and you want to read them in as separate spool files (or you want to spool them to different userids), you must separate the cards and read the decks onto disk individually. The CP system, after reading an ID card, continues reading until it reaches the end of the deck of cards.

Sending Files to Other Users

You can use the SENDFILE command to send a copy of a file to the virtual card reader of another user. To use the SENDFILE command, you specify the name of the file and the userid:

```
sendfile prog6 assemble a henry
```

Once you have sent the file, the other user can use the RECEIVE command to place the file on their A-disk.

Using Your Card Punch

You can use the CMS PUNCH command to create a punched copy of a CMS file. Once you use the CMS PUNCH command to punch a file, a READ control card is punched to precede the deck, so that the card deck can be identified. If you do not wish to punch a READ control card (also referred to as a header card), you can use the NOHEADER option on the PUNCH command:

```
punch prog8 assemble * (noheader
```

You should use the NOHEADER option whenever you punch a file that is not going to be read by the RECEIVE command.

The PUNCH command can only punch records of up to 80 characters in length. If you need to punch a file that has more than 80 characters, you can use the DISK DUMP command:

```
disk dump prog9 data
```

The RECEIVE command can also be used to read a file that has been punched using the DISK DUMP command:

```
receive = prog6 assemble
```

Using the MOVEFILE Command

You can use the MOVEFILE command, in conjunction with the FILEDEF command, to place a file in your virtual reader, or to copy a file from your reader to another device. For example:

```
spool punch to *
filedef output punch
filedef input disk coffee exec a1
movefile input output
```

The file COFFEE EXEC A1 is punched to your virtual card punch (in card-image format) and spooled to your own virtual reader.

Creating Files Using Your Punch

Apart from the procedures shown above that transfer whole files with one or two commands, there are other methods you can use to create files using your virtual punch. From a program or an EXEC file, you can punch one line at a time to your virtual punch. Then use the CLOSE command to close the spool file:

```
close punch
```

Depending on how the punch was spooled (the TO setting), the virtual punch file is either punched or transferred to a virtual reader.

Punching Cards Using I/O Macros:

If you write an OS, DOS, or CMS program that produces punched card output, you should make an appropriate file definition. If you are an OS user, you should use the FILEDEF command to define the punch as an output data device; if you are a DOS user, you must use the ASSGN command. If you are using the CMS PUNCHC macro, the punch is assigned for you. The spooling characteristics of your virtual punch control the destination of the punched output.

Punching Cards From an EXEC:

In a System Product Interpreter, EXEC 2, or CMS EXEC, you can use the CMS commands EXECIO, PUNCH and DISK DUMP to punch CMS files.

Handling Tape Files in CMS

There are a variety of tape functions that you can perform in CMS, and a number of commands that you can use to control tape operations or to read or write tape files. One of the advantages of placing files on tapes is portability: it is a convenient method of transferring data from one real computing system to another. In CMS, you can use tapes created under other operating systems. There are also two CMS commands, TAPE and DDR, that create tape files in formats unique to CMS, that you can use to back up minidisks or to archive or transfer CMS files.

Under VM/SP, virtual addresses 181 through 187 and 288 through 28F are usually reserved for tape devices. In most cases, you can refer to these tapes in CMS by using the symbolic names TAP0 through TAPF. In any event, before you can use a tape, you must have it mounted and attached to your virtual machine by the system operator. When the tape is attached, you receive a message. For example, if the operator attaches a tape to your virtual machine at virtual address 181, you receive the message:

```
TAPE 181 ATTACHED
```

The various types of tape files, and the commands and programs you can use to read or write them are:

TAPE Command

The CMS TAPE command creates tape files from CMS disk files. They are in a special format, and should only be read by the CMS TAPE LOAD command. For examples of TAPE command operands and options, see the section entitled "Using the CMS TAPE Command" on page 127.

TAPPDS Command:

The TAPPDS command creates CMS disk files from OS or DOS sequential tape files, or from OS partitioned data sets.

TAPEMAC Command:

The TAPEMAC command creates CMS MACLIB files from OS macro libraries that were unloaded onto tape with the IEHMOVE utility program.

MOVEFILE Command:

The MOVEFILE command can copy a sequential tape file onto disk or a disk file onto tape. It can move files from your reader to tape or from tape to your punch.

User Programs:

You can write programs that read or write sequential tape files using OS, DOS, or CMS macros.

Access Method Services:

Tapes created by the EXPORT function of access method services can be read only using the access method services IMPORT function. Both the IMPORT and EXPORT functions can be invoked in CMS using the AMSERV command. The access method services REPRO function can also be used to copy sequential tape files.

DDR Program:

The DDR program, invoked with the CMS command DDR, dumps the contents of a virtual disk onto tape, and should be used to restore such files to disk. Specifying the COMPACT option on the DDR command will put tape output in a compact format that uses less tape space than standard, non-compact format. Tapes in compact format may then be used as input to DDR.

Using the CMS TAPE Command

The CMS TAPE command provides a variety of tape handling functions. It allows you to selectively dump or load CMS files to and from tapes, as well as to position, rewind, and scan the contents of tapes. You can use the TAPE command to save the contents of CMS disk files, or to transfer them from one VM/SP system to another. The following example shows how to create a CMS tape with three tape files on it, each containing one or more CMS files, and then shows how you, or another user, might use the tape at a later time.

The example is in the form of a terminal session and shows, in the "Terminal Display" column, the commands and responses you might see.

System messages and responses are in black type, and user-entered commands are in blue. The "Comments" column provides explanations of the commands and responses.

Terminal Display / **Comments**

```
Tape 181 attached
```

Message indicates that the tape is attached.

```
listfile * assemble a (exec
Ready;
cms tape dump
TAPE DUMP PROG1 ASSEMBLE A1
```

Prepare to dump all ASSEMBLE files by using the LISTFILE command EXEC option; then execute the CMS EXEC using TAPE and DUMP as arguments.

```
Dumping.....
PROG1    ASSEMBLE A1
TAPE DUMP PROG2 ASSEMBLE A1
Dumping.....
PROG2    ASSEMBLE A1
TAPE DUMP PROG3 ASSEMBLE A1
.
.
.
```

The TAPE command responds to each TAPE DUMP by printing the file identification of the file being dumped.

```
TAPE DUMP PROG9 ASSEMBLE A1
Dumping.....
PROG9    ASSEMBLE A1
Ready;
```

The last file, PROG9 ASSEMBLE, is dumped.

```
tape wtm
Ready;
```

TAPE command writes a tape mark to indicate an end of file.

```
tape dump mylib maclib a
Dumping.....
MYLIB    MACLIB   A1
Ready;
tape dump cmslib maclib *
Dumping.....
CMSLIB   MACLIB   S2
Ready;
```

Two macro libraries are dumped, by specifying the file identifiers.

```
tape wtm
Ready;
```

Another tape mark is written.

```
tape dump mylib txtlib a
Dumping.....
MYLIB    TXTLIB   A1
Ready;
```

A TEXT library is dumped.

```
tape wtm 2
Ready;
```

Two tape marks are written to indicate the end of the tape.

```
tape rew
Ready;
```

The tape is rewound.

Terminal Display / **Comments**

```
tape scan (eof 4
Scanning....
PROG1    ASSEMBLE A1
PROG2    ASSEMBLE A1
PROG3    ASSEMBLE A1
PROG4    ASSEMBLE A1
PROG5    ASSEMBLE A1
PROG6    ASSEMBLE A1
PROG7    ASSEMBLE A1
PROG8    ASSEMBLE A1
PROG9    ASSEMBLE A1
```

The tape is scanned to verify that all of the files are on it.

```
End-of-file or end-of-tape
MYLIB    MACLIB   A1
CMSLIB   MACLIB   S2
End-of-file or end-of-tape
MYLIB    TXTLIB   A1
```

Tape mark indication.

```
End-of-file or end-of-tape
End-of-file or end-of-tape
Ready;
```

Two tape marks indicate the end of the tape.

```
det 181
Tape 181 Detached
```

The CP DETACH command rewinds and detaches the tape.

******* The tape created above is going to be read.*******

```
Tape 181 Attached
```

Message indicating the tape is attached.

```
tape load prog4 assemble
```

One file is to be read onto disk.

```
Loading.....
PROG4    ASSEMBLE A1
Ready;
```

The TAPE command displays the name of the file loaded. Any existing file with the same filename and filetype is erased.

```
tape scan
Scanning....
PROG5    ASSEMBLE A1
PROG6    ASSEMBLE A1
PROG7    ASSEMBLE A1
PROG8    ASSEMBLE A1
```

The remainder of the first tape file is scanned.

```
End-of-file or end-of-tape
Ready;
```

Indication of end of first tape file.

```
tape scan
Scanning....
MYLIB    MACLIB   A1
CMSLIB   MACLIB   S2
End-of-file or end-of-tape
Ready;
```

The second tape file is scanned.

Terminal Display	Comments

`tape bsf 2`
`Ready;`
	The tape is backed up and positioned in front of the last tape file.

`tape fsf`
`Ready;`
	The tape is forward spaced past the tape mark.

`tape load (eof 2`
`Loading.....`
`MYLIB    MACLIB   A1`
`CMSLIB   MACLIB   A2`
`End-of-file or end-of-tape`
`MYLIB    TXTLIB   A1`
`End-of-file or end-of-tape`
`Ready;`
	The next two tape files are going to be read.

`detach 181`
`Tape 181 detached`
	The tape is detached.

Tape Labels in CMS

Support in the CMS component of VM/SP to process labelled tapes includes the following features:

- Checks IBM standard labels on input.
- Writes IBM standard labels on output.
- Allows you to specify routines to process standard user labels during DOS and OS macro simulation under CMS.
- Allows you to specify exits for processing tapes with nonstandard labels during execution of CMS macro simulations and some CMS tape operation commands. CMS processes all tape labels; CP does not process tape labels.

Limitations

CMS tape label processing does not include:

- Label processing for tapes that are read backwards
- Processing of multi-volume files on tapes, except under OS simulation for standard labels
- Support for ANSI tapes or ASCII labels
- Label processing for any functions of the CMS TAPE command except the two functions DVOL1 and WVOL1 that process VOL1 labels.

User Responsibilities

You must initiate all your own tape label processing. To specify that you have a labelled tape, use the FILEDEF command for an OS simulation program, or use a DOS DTFMT macro for a CMS/DOS program. You can also use the TAPESL macro to process standard HDR1 and EOF1 labels and the CMS TAPE command to write and display standard VOL1 labels. You can provide IBM standard label description details with the LABELDEF command for all types of label processing. After label processing has been requested, it occurs automatically and there is no interaction between you and CMS unless an error occurs. See the "Error Processing" section later in this publication for a discussion of error processing.

Label Processing in OS Simulation

If you are running an OS simulation program and using OPEN and CLOSE macros, you specify the type of label processing you want in a FILEDEF command for a given file. Detailed information about the FILEDEF command is found in the *VM/SP CMS Command Reference.* You may specify that you want standard label processing (with SL) or nonstandard label processing (with NSL). If you choose nonstandard label processing, you must already have written a routine to process nonstandard labels. The name of this routine must be specified by the filename in the NSL parameter on FILEDEF. An example of nonstandard label processing is given in the section "NSL Processing." To be sure that the tape you are using contains no IBM labels, you may specify no label processing (NL) in the FILEDEF command. When NL is specified, CMS does not open files on a tape containing a VOL1 label as its first record. You also can specify bypass tape label processing (BLP) on a FILEDEF command. BLP tells CMS to bypass tape label processing for a file, and instead, to position the tape at a particular file before processing the data records in the file. If you specify LABOFF for a FILEDEF tape file, label processing is turned off and there is no tape positioning or label checking.

LABOFF is the default, so you do not receive any processing or tape positioning for a tape file unless you specifically request it. If you specify BLP, NL, SL, or SUL processing but omit a positional parameter, the position defaults to 1 and the tape is positioned at the first file. Examples of NL, BLP, and LABOFF processing are given in the sections "No Label (NL) Processing," "Bypass Label (BLP) Processing," and "Label Off (LABOFF) Processing."

IBM Standard Tape Label Processing

For IBM standard labels, you specify, SL or SUL, and optional positional and VOLID parameters. On a FILEDEF command, SUL means standard user labels. Everything you do for SL files, you must also do for SUL files. The positional parameter for standard label files works the same way it does in OS/VS. If you specify:

```
filedef filex tap1 sl 2
```

the tape is spaced to what is physically the fourth file on the tape before processing begins. The reason for this spacing is that a standard labelled tape has one header file, one data file, and one trailer file for each data file. If you leave off the positional parameter:

```
filedef filey tap3 sul
```

you get the first file on the tape.

The optional VOLID parameter on the FILEDEF command allows you to specify the volume serial number in the VOL1 label of a tape in case you want only the VOL1 label checked on the tape. If you want to specify other fields in IBM standard labels, you must also provide a LABELDEF statement for the tape file. The LABELDEF statement allows you to assign values to all fields in a standard HDR1 or EOF1 label. A complete description of how the LABELDEF command works may be found in the section entitled "LABELDEF Command" on page 145.

The following command defines filez as a standard labelled tape file on a tape with a VOL1 label and a volume serial number of DEPT78:

```
filedef filez tap1 sl volid dept78
```

If you also wish to specify a data set identifier for filez, you must furnish a LABELDEF command for filez as well as the FILEDEF command. Data set name may not be specified on the FILEDEF command. The LABELDEF statement below assigns a data set name of payroll to filez.

```
labeldef filez fid payroll
```

You can also specify file sequence number, volume sequence number, expiration date and other fields on a LABELDEF command. However, if you are using OS simulation macros (OPEN, CLOSE, READ, WRITE, GET, PUT, etc.) to process your tape file, the only LABELDEF parameter that has meaning for input files is fid (data set identifier). This field and the VOLIDs are checked on input by OS simulation. The other LABELDEF fields are used to specify values to be written in output labels. They are also used by other types of tape label processing (CMS/DOS and CMS) to check input labels. If no LABELDEF command has been supplied for output files, default values are used to write out labels (see the section on the LABELDEF command for the default values).

After you have set up your descriptive information for a standard labelled tape file in FILEDEF and LABELDEF statements, you run a regular OS simulation program under CMS. During program execution, HDR1 and HDR2 labels are written or checked at OPEN time. EOF1 and EOF2 labels are written or checked at CLOSE time. To have EOF labels processed, you must issue a CLOSE macro.

The VOL1 label on a tape is checked whenever a file on that tape is opened if the user has specified a VOLID parameter on his FILEDEF statement or LABELDEF statement for the file. If the volid is specified on both

LABELDEF and FILEDEF, the more recent specification is used. If no volid is specified, it is not checked.

After checking the volid, the tape is positioned and the HDR label is processed. For processing multifile volumes, you may wish to use the LEAVE option on the FILEDEF command. This option prevents a tape from being rewound and positioned before each tape file is processed. The LEAVE option does not exist on an OS DD statement. To process multi-volume files, specify the 'VOLID ?' operand on the LABELDEF command. You will be prompted for the list of serial numbers needed to process the file.

For input files, the EOF2 label is skipped. The information from the HDR2 label is merged with the FCB information and then with the DCB information. This merged information does not overlay existing data. Only the empty fields are filled with the information from the HDR2 label and the FCB. Output HDR2/EOF2 records are written from information in the DCB and the CMSCB (FCBSECT). Note that the tape density and TRTCH fields in HDR2/EOF2 records are taken from what the user specifies in his FILEDEF command for the tape file. They may not correspond to the actual density and TRTCH fields used to write the tape.

When processing standard labelled tapes in OS simulation, the following applies:

1. Multivolume tape processing will not occur if you issue the TEOVEXIT macro.

2. Multivolume tape processing only applies to CMS OS QSAM simulation.

3. During end-of-volume processing, the NOEOV operand of the FILEDEF command is ignored.

4. The VOLSEQ operand of the LABELDEF command is ignored.

5. Existing VOL1 labels are automatically rewritten for density incompatibility. However, VOL2 - VOLn, and user header labels are not rewritten.

To process standard user labels in OS simulation, you must do the following:

1. Specify the file as SUL in a FILEDEF command.

2. Provide a routine to process the user standard labels in your program.

3. Put the address of the user label routine in the DCB EXIT list of the DCB for the file. See the IBM publication *OS/VS1 Data Management Services Guide* or *OS/VS2 MVS Data Management Services Guide*, for instructions on how to establish a DCB EXIT list, and the exact linkage for communication between user label routines and the operating

system. This exact linkage should be used under CMS with the following exceptions:

a. There is no support for code x'06' EOV EXIT routine.

b. For input labels, return codes 8 and 12 from the user routine are not supported. If an input return code is not 0, it is treated as if it were 4.

4. Note that your standard user label routines do not perform any input/output. They set up an output label for writing, but the CMS tape label processing routines actually write out the label. For input, the CMS label processing routines read in your user standard label but then give control to your routine to check the label.

No Label (NL) Processing

You should specify NL in the FILEDEF command when you expect a tape does not contain any IBM standard tape labels. CMS reads your tape at the time a file is opened and does not open the file if the tape contains a VOL1 label as its first record. If the tape does not contain a VOL1 label, a file is opened and the tape is positioned by using the position parameter (n). For example, if you specify:

```
filedef fileq tapl nl 2
```

fileq is not opened if the tape on tapl (181) has a VOL1 label. If the tape does not have a VOL1 label, fileq is opened and the tape is positioned at the second file. If you do not specify a position parameter, the tape is positioned at the first file, (that is, the load point).

Bypass Label (BLP) Processing

You should specify BLP in the FILEDEF command to bypass tape label processing. CMS does not check your tape for an IBM standard tape label. It uses the position parameter you specified to position the tape during open processing. If you do not specify a position parameter, the default is 1. For example:

```
filedef fileabc tapl blp 4
```

positions the tape at the fourth file when it opens fileabc. Because CMS does not know whether files on the tape are label files or data files, the tape is positioned at what is physically the fourth file, regardless of file content. Any label files on the tape are included in counting files.

Label Off (LABOFF) Processing

You should specify LABOFF in the FILEDEF command if you want no positioning or label processing to occur during open processing. The position parameter is not valid for LABOFF. If you specify LABOFF, and your tape is positioned at record 6 in the third file before you issue an OPEN macro, the tape is positioned at exactly the same record after open processing (record 6 in the third file). The following FILEDEF command does not move tape2 (182) before processing the data in fileb:

```
filedef fileb tap2 laboff
```

Nonstandard Label (NSL) Processing

In order to process nonstandard labels, you must write your own routine to read, write, and check the labels. If you have such a routine as a CMS TEXT or MODULE file, you put the filename of the routine after the NSL keyword parameter in the FILEDEF command for the file. The filename must be the name of the first CSECT in the program. It is to this point that control is transferred when the NSL routine gets control. If you do not have a TEXT or MODULE file with the NSL filename you specify, you get an error message. The OPEN and CLOSE routines will load your module if it is not already in storage and will pass control to it at the time they are opening or closing the file. Your routines will then be responsible for processing the tape labels. Nonstandard label routines must do the actual reading and writing of tape labels as well as checking and setting up the label. This is one of several ways nonstandard label processing is different from standard user label processing. Because the CMS label processing routines do not know the size or format of your nonstandard labels, they cannot read or write the labels.

If you use a MODULE file for an NSL routine, it is important that you create the MODULE file so that it starts at an address that will not allow it to overlay the program or command you are executing at the time the NSL routine is invoked. The reason for this restriction is that the NSL routine is dynamically loaded while your program is executing. For the TAPEMAC and TAPPDS commands, starting the NSL routine at an address above X‘21000’ prevents such an overlay. If the NSL routine is invoked from your own program which is running in the user area, you must determine how big your program is and where the NSL MODULE file should be located to prevent overlay. Note that you do not have to specify a starting address for NSL routines that are TEXT files. The CMS loader loads such files for you at an address that does not cause an overlay.

Although any user may write his own NSL routine, it is expected that a system programmer will usually write such routines and then other programmers in the installation will use them. Before writing an NSL routine, refer to *VM/SP CMS for System Programming* for details on interrupt handling, storage, supervisor calls, and related information. In order to ensure proper communication with the CMS system routines, you must use the linkage described below when you write nonstandard label routines.

When an NSL tape label processing routine gets control, register 1 points to a 16-byte parameter list with the following format:

byte 0	Type call	Caller id	Tape Mode-Set Byte	Reserved	
byte 4	TAPID				
byte 8	FCBSECT address				ID parameter for TAPEMAC and TAPPDS
byte 12	DCB address				

The Type call field is a code telling the type of label processing being done:

```
x'00'     is OPEN input
x'04'     is OPEN output
x'08'     is CLOSE input
x'0C'     is CLOSE output
x'10'     is End Of Tape output
```

The Caller id is a one-byte code which is one of the following:

```
x'80'      Call by OS simulation
x'20'      Call by CMS TAPEMAC or TAPPDS commands
```

Tape modeset byte is used to communicate with the CMS tape I/O routines. It is a one byte hexadecimal code that depends on the type of tape (7, 9, or 18 track), tape density, etc. For further information on the Mode Set, see the TAPE command description in the *VM/SP CMS Command Reference* (You probably will pass this byte to the CMS tape controlling module to read and write your tape labels and will never need to know what its codes mean.)

FCBSECT address is the address of the CMSCB (FCBSECT) for the tape file you are processing.

DCB address is the address of the DCB for the tape file you are processing.

Note: For the TAPEMAC and TAPPDS commands, the same interface is used, except that instead of the FCBSECT and DCB address fields, the eight character identifier specified in the ID=identifier field in the command is passed. This identifier enables you to identify which file you are processing since the TAPEMAC and TAPPDS commands do not work with CMSCBs or DCBs.

Control is passed to your NSL routine by a BALR 14,15 instruction so register 15 contains the address of your routine when you receive control. Register 14 contains the address you should return to when you are finished processing the nonstandard labels. You can return with a BR 14 instruction. When you receive control, register 13 points to a save area in which to store the callers register. The save area linkage is standard OS/VS linkage. You receive control with a PSW key of X'E' which allows you to modify only user storage. When you are finished processing, place a code in register 15 to the CMS label processing routine that called your routine. Place the value 0 (zero) in register 15 if there have been no errors and you want processing to continue normally and the data set to be opened. If you return a nonzero value in register 15, a message is issued to your terminal and the data set is not opened.

If you write the following FILEDEF statement:

```
filedef tapfl tapl nsl readlab
```

and have a program called READLAB as a MODULE or TEXT file, your program will receive control when the data set called tapfl is opened. When your program gets control, register 1 contains the address of the parameter list described above. Using the data in this parameter list, you are able to read or write your own tape header labels. When the same data set is closed, your program again receives control and you can read or write your own trailer labels. Your program can test whether it is getting control for OPEN or CLOSE by examining the type call byte in the parameter list passed to you. If the type call byte is x'10', your NSL routine is being invoked while you are writing an output data set and you have reached the reflective mark that indicates end of tape. You may wish to do special processing in this case. See the "End of Tape" and "End of Volume" section in this publication for further information on end of tape processing.

Differences Between Tape Label Processing Under OS/VS and OS Simulation in CMS

There are a few minor differences in the way CMS OS simulation processes tapes and the way OS/VS processes them. These differences are listed below.

- If you are using OS/VS and you do not specify any label parameter on your JCL statement, the default is SL or standard labels. When you use OS simulation under CMS and do not specify any label information on a FILEDEF statement, the default is LABOFF. LABOFF turns off label processing and nothing is done to position the tape or process labels. Thus, if you specify no label information on FILEDEF, the system will process your tape files exactly the same way they are processed on a CMS system that has no tape label processing facilities.

- You must specify CLOSE to process all trailer labels. No automatic CLOSE occurs at end of data or after reading a tape mark. There is no EOV monitor to process labels before a data set is closed. If an input tape is positioned at an EOF1 or EOV1 record when CLOSE is issued, the label is processed. If a tape file is closed before all data records are read, the trailer label is not processed. Output tapes have EOF records written only at CLOSE time.

- There is no deferred label processing under OS simulation in CMS.

- When the user has not specified a block count routine in his DCB EXIT list under OS/VS, the program abends when a block count error occurs. Under CMS, this condition produces a message that asks whether or not to abend the operation.

- Certain fields in HDR1 and EOF1 labels default to values different from those under OS/VS. These values can always be specified in a LABELDEF command if the user does not like the default values. For example, the default for data set name in an output label under OS simulation is DDNAME and not DSNAME. The default data set sequence number is always one even when the data set is not the first data set on the tape. The default volume sequence number is always one. Read the section entitled "LABELDEF Command" on page 145 to learn what the default values are under CMS. You can find what default values are in OS/VS by reading the IBM publication *OS/VS Tape Labels.*

  Note: You can always get exactly what you want written on a tape label by explicitly specifying the field on a LABELDEF command. For example, you can specify DSNAME as FID on such a command and have it written in the label instead of DDNAME.

- Default volids (when you do not specify a volid in a LABELDEF or FILEDEF statement) in output HDR1 and EOF1 records under CMS will be CMS001 and will not be the actual volume serial in the VOL1 record already on the tape, unless you are processing 'SL' tapes, in which case it will be the actual volume serial already on the tape. It is recommended that you always specify the volid in FILEDEF or LABELDEF to be sure the information written is correct.

- Expiration date specification is always done in absolute form rather than by retention period. You must always use the form yyddd where yy is the year (0-99) and ddd the day (0-366). CMS does not handle expiration dates specified by retention periods.

- When CMS reads a HDR1 label and finds an unexpired file, it always issues a message allowing you to enter "ERROR" or "IGNORE." "ERROR" prevents opening the file in OS simulation. When the DISP MOD option of the FILEDEF command is specified for SL tapes, "IGNORE" allows you to have the tape positioned at the end of the file, ready to add new records. Otherwise, "IGNORE" causes the existing record to be overwritten.

- The NSL routine linkage is quite different under CMS than in OS/VS. (See the section "NSL Processing" for details.)

- Volume serial number verification occurs every time a file on a tape is opened under OS simulation unless the FILEDEF LEAVE option is used for multifile tapes.

- Existing VOL1 labels are automatically rewritten for density incompatibility in CMS as they are in OS/VS. However, VOL2 - VOLn and user header labels are not rewritten.

- The information from the HDR2 label is checked and merged with the FCB information and then with the DCB information. The merged information does not overlay existing data. Only the empty fields are filled with the information from the HDR2 label or the FCB.

- To maintain OS compatibility in the EOV2/EOF2 label, you must specify LRECL in the output FILEDEF.

- Multivolume tape processing only applies to CMS OS QSAM simulation.

- Blank tapes used for output in CMS cause the tape to run off the reel if you define the tape file as SL or NL. The tape label processing routines try to read an existing VOL1 or HDR1 label before writing on the tape. Therefore, you should always use the CMS TAPE command to write at least one tape mark (for NL tapes) or a VOL1 label (for SL or SUL tapes) before using the tape to write an output data set.

- If you specify a position parameter that is too big (that is, there are not that many files on the tape), the tape will run off the reel in CMS.

- There are no user exits for user standard labels for EOV label processing in CMS.

- CMS does not support user return codes of 8 and 12 for input standard user labels. If the return code from a user routine is not zero after input label processing, CMS treats it as if the return code was 4. (See the IBM publication *OS/VS1 Data Management Services Guide* or *OS/VS2 MVS Data Management Services Guide* for details).

- No count is kept of user standard labels read or bypassed in CMS. If more than eight such labels exist, the fact is not detected.

- User label processing routines do not receive control under CMS when an abend or a permanent I/O error occurs.

- If a CMS output tape is not positioned at a HDR1 label or a tape mark when label processing begins, error message 422 is issued. Under OS/VS such conditions cause an abend.

- TCLOSE with the REREAD option causes a tape to be rewound under CMS and then forward spaced one file if the tape has standard labels. Under OS/VS, the tape is backspaced four files and forward spaced one file. REREAD for unlabeled tapes in CMS always causes a rewind.

For further information on OS/VS tape label processing, refer to the following IBM publications: *OS/VS1 Data Management Services Guide*, *OS/VS2 MVS Data Management Services Guide*, and *OS/VS Tape Labels*.

For details on end-of-tape/end-of-volume processing under CMS, refer to "End-of-Volume and End-of-Tape Processing" on page 146.

Label Processing in CMS/DOS

You specify the type of label processing you want in CMS/DOS on a DTFMT macro in exactly the same way you specify it when you want to run your program under VSE. See *VM/SP CMS for System Programming* for details on CMS support for the DTFMT macro.

Labelled tapes are only supported if you use the DTFMT macro. There is no support for labelled tapes in CMS/DOS for any other type. If you try to read labelled tapes with a DTFCP or DTFDI macro, input standard IBM header labels are skipped, but no other input labels are processed. Output tapes with standard labels have these labels overwritten with a tape mark. All tape work files are treated as output unlabeled files in CMS/DOS although they are defined by a DTFMT. Tapes used for such files have a tape mark written as the first record when the file is opened.

Unlabeled and Nonstandard Labelled Tapes

You define an unlabeled tape with the DTFMT parameter FILABL=NO. The tape file is processed as having no labels.

You define a nonstandard labelled tape with the DTFMT parameter FILABL=NSTD. You also must provide a routine to process your nonstandard labels in the LABADDR=parameter of the DTFMT. Tape processing in CMS for these files is the same as it is under VSE.

Standard Labelled Tapes

You define a standard label tape with the DTFMT parameter FILABL=STD. You also must supply a LABELDEF command to specify label description information. This command replaces the VSE TLBL card and is required for standard label processing under CMS/DOS. Refer to the section entitled "LABELDEF Command" on page 145.

In order to connect the LABELDEF command for a file with the DTFMT for the same file, you must use the same name to label your DTFMT as you use for a filename in your LABELDEF command. If you code a DTFMT macro in your program as:

```
MT1 DTFMT      ...FILABL=STD
```

you must then supply the following type of LABELDEF command:

```
labeldef mt1 fid yourfile fseq...
```

You can put any description parameters you want on your LABELDEF command but the filename for it must be mt1 if you coded MT1 as the label on the DTFMT.

After you have set up your DTFMT and LABELDEF, you execute your CMS/DOS program. HDR1 labels are checked or written when an OPEN macro is issued. EOF1 labels are checked or written when a CLOSE macro is issued. A VOL1 label volume serial number is checked only if the tape is positioned at load point when the label processing begins and if you have specified a VOLID parameter on a LABELDEF statement for the file. Note, if NOREWIND is not specified in the DTFMT macro for the file, the tape is rewound so it is positioned at load point for label processing.

If you want to process user standard labels as well as standard labels in CMS/DOS, you specify FILABL=STD and also supply a LABADDR parameter in the DTFMT for the file. Control is then transferred to your label processing routines after standard labels are processed. The linkage to user standard label routines is exactly the same as in VSE.

Differences Between Tape Label Processing Under VSE and CMS/DOS

There are minor differences in the way tapes are processed by CMS/DOS and the way they are processed by VSE. These differences are:

- The tape error messages are CMS error messages and not VSE error messages. In some cases VSE allows the system operator to reply NEWTAP to an error message. The system then waits for the operator to mount a new tape and continues processing with this new tape. Such a reply is never possible under CMS/DOS. In CMS/DOS, you usually can reply IGNORE to ignore a tape label error condition or CANCEL to cancel a job. NEWTAP is never allowed. In a few cases, CMS/DOS allows an IGNORE reply where VSE does not.

- You must specify CLOSE to process all trailer labels. No automatic CLOSE occurs at end of data or after reading a tape mark. If an input tape is positioned at an EOF1 or EOV1 record when CLOSE is issued, the label is processed. If a tape file is closed before all data records are read, the trailer label is not processed. Output tapes have EOF records written only at CLOSE time. For nonstandard labelled tapes, your own routines do not receive control on input when a tape mark is read. You must issue a CLOSE macro in your EOFADDR routine in order to have the trailer labels processed.

- Certain fields in HDR1 and EOF1 labels default to values different from those in VSE. For example, the default volume serial number written in a HDR1 label is CMS001 and not the actual volume serial number (volid) in the VOL1 label already on the tape. The default file sequence and volume sequence numbers are always one even when the file is not the first file on the tape. You should read the section entitled "LABELDEF Command" on page 145 in this publication to learn what the default values are in CMS/DOS. You also can read the IBM publication *VSE/AF Tape Labels* to find what the default values are for VSE. If you do not like the default values, you can always specify the exact values you want in label fields in a LABELDEF command.

- Expiration date specification is always done in absolute form rather than by retention period. You must always use the form yyddd where yy is the year (0-99) and the ddd the day (0-366). CMS does not handle expiration dates specified by retention periods.

- VOL1 labels written in the wrong density are not rewritten automatically by CMS/DOS as they are by VSE.

- Blank tapes should not be used for tape files specified as FILABL=STD in CMS/DOS; they will run off the reel. Use the CMS TAPE command to write a VOL1 label or a tape mark on a blank tape before using it for a STD file.

- Not all tape movement and label checking that occurs in VSE occurs under CMS. For example, when opening an output file, a VSE system expects the tape to be positioned at a HDR1 label or a tape mark. It then backspaces the tape to read the last EOF1 label on the tape. If it does not find the label it expects, it issues an error message. This check is not performed by CMS/DOS. If the tape is not positioned at a HDR1 label or a tape mark when output open processing begins, error message 422 is issued.

- After an EOV1 label is written (see "End-of-Volume and End-of-Tape Processing" on page 146), the tape is always rewound and unloaded under CMS/DOS. VSE lets a DTFMT parameter control whether or not the tape is rewound.

- User label processing routines do not receive control when an I/O error occurs under CMS/DOS.

- Control is not passed to user standard label routines in CMS/DOS when EOT has been sensed on output and an EOV1 label has been written by the system routines.

- Work tapes are not checked for an expiration date when they contain standard labels under CMS/DOS. If a tape is to be opened as a work tape, CMS/DOS tests to see if it contains a VOL1 label. If it does, a dummy HDR1 label and a tape mark are immediately written on the tape after the VOL1 label. If the tape does not contain a VOL1 label, a tape mark is written at the beginning of the tape. VSE checks expiration dates on previously labelled tapes used as work tapes and gives the operator a chance to reject the tapes if the expiration date has not expired.

For further information on VSE and CMS/DOS tape label processing, refer to the IBM publications, *VSE/AF Tape Labels* and *VSE/AF Macro User's Guide.*

CMS TAPESL Macro

The TAPESL macro is provided for use in CMS programs that do not use OS and DOS simulation features. You can use the CMS TAPESL macro to process IBM standard HDR1 and EOF1 labels without using DOS or OS OPEN and CLOSE macros. You will probably use TAPESL with the RDTAPE, WRTAPE, and TAPECTL macros.

TAPESL processes only HDR1 and EOF1 labels. It does not perform any functions of opening a tape file other than label checking or writing. The TAPESL macro generates linkage to the CMS tape label processing routine that actually processes the label. The macro generates a block of data (32 bytes long) in order to communicate with the tape label processing routines. TAPESL is used both to check and to write tape labels. A LABELDEF command must be issued prior to running the program that contains this macro. The LABID parameter of the TAPESL macro is used to specify the name of the LABELDEF to be used. For example, if you use the macro:

```
TAPESL HOUT,181,LABID=GOODLAB
```

in your assembly language program, you must supply a LABELDEF command for GOODLAB:

```
labedef goodlab fid file10 fseq 4 exdte 78235
```

The tape must be positioned correctly (at the label to be checked or at the place where the label is to be written), before you issue the macro. TAPECTL may be used to position the tape. TAPESL reads or writes only one tape record unless you specify SPACE=YES for input. Then it spaces the tape to beyond the tape mark that ends the label file. TAPESL reads and checks a tape VOL1 label provided the tape is positioned at load point and the user has specified a volid in his LABELDEF command.

Tape Label Processing by CMS Commands

There are three types of CMS commands that do some type of tape label processing. They are:

- TAPEMAC and TAPPDS commands
- TAPE command
- MOVEFILE command.

TAPEMAC and TAPPDS Commands

TAPEMAC and TAPPDS have operands where you can indicate the type of label processing you want. The tape must be positioned properly (at the data file or label file you want) before you issue the command. The TAPE command may be used for positioning. A separate LABELDEF command is required for these commands if IBM standard label checking is desired. If SL label type is specified without a labdefid, standard header labels are displayed on the terminal but not checked by the CMS label processing routines. The command:

```
tapemac macfile SL (tap2
```

displays any standard labels that exist on your terminal while the series of commands:

```
labeldef maclab fid macro volseq 2 crdte 77102
tapemac macfile sl maclab (tap2
```

invokes the CMS tape label processing routines. These routines check to see that your tape has a HDR1 label that has a file identifier of macro, a volume sequence number 2, and a creation date of 77102. VOL1 labels are not checked during label processing by TAPEMAC and TAPPDS unless the tape is positioned at load point and you have specified a volid on your LABELDEF command. The DVOL1 function of the TAPE command can be used for volume verification before positioning the tape if the user does not want to start at the first file. These commands process only HDR1 labels; they skip HDR2, UHL, and all trailer labels without processing them.

To process nonstandard tape labels with TAPEMAC and TAPPDS, you use the same interface described in the section "NSL Processing under OS Simulation." The only difference is that instead of putting the CMSCB and DCB addresses in the parameter list, the ID parameter you placed in the command line is passed to your NSL routine.

```
tappds pdsfile cmsut1 * nsl superck id XYZ12345
```

passes the EBCDIC identifier XYZ12345 to your nonstandard label checking routine called SUPERCK. This identifier may be up to eight characters long and is left justified in bytes 8-15 of the parameter list. You can use the identifier to inform your NSL routine of what file you are processing.

Tape Command DVOL1 and WVOL1 Functions

Use the DVOL1 function of the TAPE command to display the VOL1 label of a tape on your terminal. You may use this command to ensure the system operator has mounted the correct tape before you begin processing the tape. If the tape does not have a VOL1 label and you issue the TAPE command, you are informed that the VOL1 label is missing. Do not use TAPE DVOL1 if you have a blank tape. If TAPE DVOL1 is issued and a blank tape is used, CMS will search the entire tape to find the label record; since the tape is void of any records, the tape will run off the end of the reel.

Use the WVOL1 function on the TAPE command to write a VOL1 label on a tape. You can specify a one- to six-character volume serial number (volid) through this command and also a one- to eight-character owner field.

MOVEFILE Command

You can use the MOVEFILE command to move labelled tape files if these files are defined as labelled by the FILEDEF command. The MOVEFILE command supports only SL, NSL, BLP, NL, and LABOFF processing. SUL files are processed as SL files and no user exits are taken.

You can also use the MOVEFILE command to display tape labels on your terminal if you want to see what these labels look like. The following sequence displays the VOL1 and first HDR1 labels on tap4 if the tape has standard labels:

```
filedef in tap4
filedef out term
tape rew (tap4
move in out
```

LABELDEF Command

The LABELDEF command is used to specify the exact data you want written in certain fields of a HDR1 or EOF1 tape label for output. It can also be used to specify fields in the same labels that you want checked on input. If you do not explicitly specify a field for output, a default value is used. If you do not explicitly specify a field for input, the field is not checked. For example:

```
labeldef abc fid master volseq 1 exdte 77364
```

used for input tells CMS to check the file identifier volume sequence number and expiration date in an input HDR1 label. No other fields in the label are checked. The same specification used for output causes the HDR1 label to have MASTER written in the file identifier field, 1 written in the volume sequence number field and 77364 written in the expiration date field. Default values are written in the HDR1 fields that are not specified.

Default values for HDR1 labels are as follows:

FID	For OS simulation, the DDNAME (Specified on FILEDEF) For CMS/DOS, the DTFMT symbolic name For CMS TAPESL macro, the LABELDEF id (LABID = labeldefid) parameter
VOLID	CMS001 For OS simulation, the actual VOLID from the tape mounted is used if processing an 'SL' tape file.
VOLSEQ	0001
FSEQ	0001
GENN	Blanks

GENV	Blanks
CRDTE	Current date that label is written
EXDTE	Current date that label is written
SEC	0

The filename on the LABELDEF command is used to connect your label definition to a file defined elsewhere. This is why you specify different data for file name depending on the type of tape label processing you are doing. Filename is DDNAME for OS simulation, DTFMT symbolic name for CMS/DOS and labeldefid for TAPESL.

The LABELDEF command takes the place of the VSE TLBL statement for CMS/DOS.

End-of-Volume and End-of-Tape Processing

There is no true end-of-volume support available with CMS tape label processing, except under OS simulation for standard labelled tapes. Under OS simulation, automatic volume switching and multi-volume files are supported, but FEOV instructions are not. The multi-volume support optionally includes the ability to switch to a different tape drive when EOV is encountered. The alternate drive is specified by the ALT option of the FILEDEF command. The following features exist to aid the IBM standard labelled tape user when an end-of-tape on output or an EOV label on input is detected. These are the only ways in which CMS supports EOV processing.

- Input - For other than OS simulation processing for standard labelled tapes, when a CLOSE macro is issued or when a TAPESL macro processes an input trailer label, a message is issued if the label read is an EOV1 label instead of an EOF1 label. The EOV1 label is then processed exactly as if it were an EOF1 label. You must request that the operator mount a new tape and reopen a file if you want to continue processing the data.

  Under OS simulation, if the record after the tape mark is an EOF1 label, the exit specified in the DCB gets control. If the record is an EOV1 label, tape volume switching occurs. If the next tape has already been mounted on an alternate drive, processing continues. Otherwise, system notifies the operator to mount the next tape. For example:

  ```
  filedef in tap2 sl
  filedef out term
  labeldef in volid ?
  ```

  When you issue the LABELDEF command, you receive the following response:

  ```
  DMSLBD441R Enter VOLID information:
  ```

Now enter the volids that you need followed by the MOVEFILE command. When you have entered all the volids that you need, then enter a null line. If you initially respond with a null line, it is treated as a volid of 'scratch'. For example, if you have two volids, hal001 and hal002, you would enter them as follows:

```
hal001 hal002
(null line)
movefile in out
```

If the last label read on tap2 is an EOV1 rather than EOF1, you will get the following message:

```
Attempting to change tape volume for DDNAME xxxxxxxx
To cancel the tape volume switch, type CANCEL
```

You can stop the tape volume switch at anytime by typing CANCEL. This notifies the tape operator that you do not want the tape mounted, and terminates the execution of the MOVEFILE command. Otherwise, the system waits for the tape operator to mount the requested tape volume and issues the following messages at five minute intervals until the volume is mounted.

Note: Your system programmer can extend the wait time to longer than five minutes by using the TVSPARMS macro. Refer to *VM/SP CMS for System Programming* for more information.

```
Message sent to userid OPERATOR:
Mount tape volume HAL002 on virtual 182 without a write ring;
Request number 1
(five minute interval)
Message sent to userid OPERATOR:
Mount tape volume HAL002 on virtual 182 without a write ring;
Request number 2
(five minute interval)
Message sent to userid OPERATOR:
Mount tape volume HAL002 on virtual 182 without a write ring;
Request number 3
(five minute interval)
Wait time for tape volume switch has almost expired; to
continue waiting, type EXTEND
```

At this point you can give the tape operator more time to mount the tape by typing 'EXTEND'. The operator receives the tape mount prompts once again, beginning with request number one. Otherwise, the following message is displayed at your terminal:

```
Message sent to userid OPERATOR:
Mount tape volume HAL002 on virtual 182 without a write ring;
Request number 4
(five minute interval; type EXTEND if you need more time)
Wait time for tape volume switch has expired; tape volume
switch for volume HAL002 on virtual 182 cancelled
```

The allotted time for the tape volume switch is over and the MOVEFILE command is terminated. If you still want to process the data, you must begin again by reissuing the initial FILEDEF and LABELDEF commands.

Notes:

1. *If you issue the TEOVEXIT macro, multi-volume tape processing will not occur.*

2. *Your system support personnel can replace the OS simulation multi-volume support described above via user exits. If the multi-volume support appears to be different on your system, see your system administrator.*

- Output - Under CMS/DOS and OS simulation processing only (that is, the processing does not occur for TAPESL or CMS commands), the following limited EOV processing occurs:

  1. Except under OS simulation, if you specify that you have an IBM standard labelled tape file, a single tape mark is written to end your data. This occurs when end-of-tape is sensed on output while you are using regular access method macros to write the file. The tape mark is written immediately after the record that caused the EOT to be sensed. Following this tape mark, CMS writes an EOV1 label and a single tape mark. It then rewinds and unloads your tape. A message is issued indicating that an EOV1 label was written.

     Under OS simulation processing for standard labelled tapes, if you are at the end of the file, an EOF1 label, an EOF2 label, and a tape mark are written. If you are at the end of a tape and you need another tape to finish writing, an EOV1 label, an EOV2 label, and a tape mark are written at the end of the first tape. Then, the tape is rewound and unloaded, a message is issued indicating that an EOV label was written, and the operator is notified to mount the next tape needed to continue writing if a tape is not already mounted on an alternate drive specified in the FILEDEF command. (See the above example for the messages that are issued.)

     If you specified nonstandard labels, instead of writing the EOV label, an exit to the nonstandard label routine you specified for the file is taken after the end-of-data tape mark is written. For BLP or NL files, only the ending tape mark is written.

  2. CMS/DOS jobs are always canceled after an EOT condition is detected on output. In order to continue processing the tape, you must have a new tape mounted, run the same job over again or run a new job and reopen the file.

  3. OS simulation programs that contain a BSAM CHECK macro cause an abend when EOT is detected, with code 001 after an error message. A BSAM program that does not use a CHECK macro has no way of detecting the EOT condition. Such a program continues

to try to write on the tape after it is rewound and unloaded. The program enters a wait state rather than continue running to a normal or abnormal completion. Therefore, you should always include a BSAM CHECK macro after the WRITE if you expect your program to reach end-of-tape. OS simulation BSAM users are also responsible for completing processing on a new tape with the same or a new job after an EOT is detected.

4. If you are a CMS/DOS user you always get the automatic output end-of-tape processing described above. However, if you are an OS simulation user and the NOEOV option was specified on your FILEDEF command, it is ignored at the end-of-volume processing. However, the program causes an abend if you use QSAM or include a BSAM CHECK macro after your WRITE macro. Without a CHECK macro, a BSAM program runs the tape off the reel when EOT is sensed and NOEOV is specified.

Error Processing

When the standard label processing routines find errors or discrepancies on tape labels, they send a message to the CMS terminal user who is processing the tape. After an error message is issued, the user can ask the system operator to mount a new tape, use the CMS TAPE command to position the tape at a different file, or respecify his label description information. If you are a terminal user and want another tape mounted, you send the system operator a message telling him what tape to mount.

Some errors cause program termination and others do not. The effect of tape label processing errors depends on both the type of error and the type of program (that is, CMS/DOS, OS simulation, CMS command, etc.) that invokes the label processing. The following are general guidelines on error handling:

- Messages identifying the error are always issued.
- Under OS simulation, tape label errors result in open errors. These errors prevent a tape file from being opened. They do not necessarily end a job. Errors in trailer labels (except block count errors) have no effect on processing.
- In CMS/DOS, the terminal user is generally given two choices: ignore the error or cancel the job. The new-tape option is not allowed.
- The CMS commands TAPEMAC and TAPPDS terminates with a non-zero return code after a tape label error.
- Certain error situations such as unexpired files and block count errors for OS simulation allow the user to ignore the error and do not cause open errors. In these cases, the user enters his decision at the terminal after he is notified of the error.

- Errors that occur during the loading of an NSL routine cause an abend (code 155 or 15A). A block count abend gives an error code of 500.

In all cases, after an error has been detected and diagnosed, you must decide what to do. You may wish to have a new tape mounted and then re-execute the command or you may want to respecify your LABELDEF description if it was incorrect. You can also use the TAPE command to space the tape to a new file if it was positioned incorrectly.

The MOVEFILE Command

The MOVEFILE command can copy sequential tape files into disk files, or sequential disk files onto tape. It can be particularly useful when you need to copy a file from a tape and you do not know the format of the tape.

To use the MOVEFILE command, you must first define the input and output files using the FILEDEF command. For example, to copy a file from a tape attached to your virtual machine at virtual address 181 to a CMS disk, you would enter:

```
filedef input tap1
filedef output disk tape file a
movefile input output
```

This sequence of commands creates a file named TAPE FILE A1. Then use CMS commands to manipulate and examine the contents of the file.

MOVEFILE can also be used to display tape labels and/or move labelled tape files. See "Tape Labels in CMS" on page 130 for details.

MOVEFILE can be used to copy data from a shared (e.g. 3420) to a non-shared tape subsystem. For more information refer to "Moving Data Between Shared and Non-Shared Tape Subsystems" on page 153.

OS Utility Programs

The CMS command TAPPDS can read OS partitioned and sequential data sets from tapes created by, or for, the IEBPTPCH, IEBUPDTE, and IEHMOVE utility programs. When you use the TAPPDS command, the OS data set is copied into a CMS disk file, or in the case of partitioned data sets, into multiple disk files.

IEBPTPCH:

Sequential or partitioned data sets created by IEBPTPCH must be unblocked for CMS to read them. If you have a tape created by this utility, each member (if the data set is partitioned) is preceded with a card that contains "MEMBER=membername." If you read this tape with the command:

```
tappds *
```

then, CMS creates a disk file from each member, using the membername for the filename and assigning a filetype of CMSUT1. If you want to assign a particular filetype, for example TEST, you could enter the command as follows:

```
tappds * test
```

If the file you are reading is a sequential data set, you should use the NOPDS option of the TAPPDS command:

```
tappds test file (nopds
```

The above command reads a sequential data set and assigns it a file identifier of TEST FILE. If you do not specify a filename or filetype, the default file identifier is TAPPDS CMSUT1.

IEBUPDTE:

Tapes in control file format created for the IEBUPDTE utility program can be read by CMS. Data sets may be blocked or unblocked, and may be either sequential or partitioned. Since files created by IEBUPDTE contain ./ADD control cards to signal the addition of members to partitioned data sets, you must use the COL1 option of the TAPPDS command. Also, you must indicate to CMS that the tape is in IEBUPDTE format. For example, to read a partitioned data set, you would enter the command:

```
tappds * test (update col1
```

The CMS disk files created are always in unblocked, 80-character format.

IEHMOVE:

OS unloaded partitioned data sets on tapes created by the IEHMOVE utility program can be read either by the TAPPDS command or by the TAPEMAC command. The TAPPDS command creates an individual CMS file from each member of the PDS.

If the PDS is a macro library, you can use the TAPEMAC command to copy it into a CMS MACLIB. A MACLIB, a CMS macro library, has a special format and can usually be created only by using the CMS MACLIB command. If you use the TAPPDS command, you have to use the MACLIB command to create the macro library from individual files containing macro definitions.

Specifying Special Tape Handling Options

For most of the tape handling that you do in CMS, you do not have to be concerned with the density or recording format of the magnetic tapes that you use. There are, however, some instances when it may be important and there are command options that you can use with the TAPE command MODESET operand and with ASSGN and FILEDEF command options.

The specific situations and the command options you should use are listed below.

- If you are reading or writing a 7-track tape and the density of the tape is either 200 or 556 bpi, you must specify DEN 200 or DEN 556.

- If you are reading or writing a 7-track tape with a density of 800 bpi, you must specify 7TRACK.

- If you are reading or writing a 7-track tape without using the data convert feature, you must use the TRTCH option.

- When using the TAPE command, if you are writing a tape using a 9-track dual density tape drive with the 9TRACK option specified, and you want the density to be 800 (on an 800/1600 drive) or 6250 (on a 1600/6250 drive), then you must specify DEN 800 or DEN 6250.

- When using the ASSGN and FILEDEF commands, if you are writing a tape using a 9-track dual density tape drive with the 9track option specified, and you want the density to be 800 (on an 800/1600 drive) or 1600 (on a 1600/6250 drive), then you must specify DEN 800 or DEN 1600.

- If you are reading or writing a tape cartridge of the 3480 Magnetic Tape Subsystem, specify the following:

TAPE command	18TRACK and/or DEN 38K
FILEDEF command	18TRACK and/or DEN 38K
RDTAPE macro	MODE=(18,38K)
TAPECTL macro	MODE=(18,38K)
TAPESL macro	MODE=(18,38K)
WRTAPE macro	MODE=(18,38K)

  Note: When using the DUMP option of the TAPE command with a 3480 device, you may also specify the tape write transfer mode, either buffered or immediate.

  ```
  TRANsfer BUFFered
  TRANsfer IMMEDiate
  ```

- If you are writing a tape, the default tape block size is 4096 bytes plus a 5-byte header. This format is not compatible with previous VM/370 systems. Therefore, if you want to write a tape compatible with previous VM/370 systems, you must use the "BLKSIZE 800" option of the TAPE command. The TAPE command is described in detail in *VM/SP CMS Command Reference.*

Moving Data Between Shared and Non-Shared Tape Subsystems

Virtual tape addresses 180 through 187 are at one virtual control unit and 288 through 28F are at a second. This allows you to copy data between a shared (e.g. 3420) and non-shared (e.g. 3480) tape subsystems. Using the FILEDEF command, assign the first tape drive to one virtual control unit (e.g. TAP1) and the other tape drive to the second virtual control unit (e.g. TAP8).

For example, these commands copy data from a non-shared device to a shared device.

```
FILEDEF IN TAP0 (18TRACK              *TAP0 is assigned to 180
FILEDEF OUT TAP8 (9TRACK              *TAP8 is assigned to 288
MOVEFILE IN OUT
```

For example, these commands copy data from a shared device to a non-shared device.

```
FILEDEF IN TAPA (18TRACK              *TAPA is assigned to 28A
FILEDEF OUT TAP1 (9TRACK              *TAP1 is assigned to 181
MOVEFILE IN OUT
```

Chapter 8. Communicating with Other Computer Users

Using CMS commands, you are able to send information (files, messages, and notes) to other computer users and to receive information from them. You can collect the necessary information about other computer users with whom you communicate to keep in your "userid NAMES" file. The following CMS commands reference the NAMES file created via the CMS NAMES command:

NAMEFIND	Display/Stack information from a NAMES file.
NOTE	Prepare a "note" for one or more computer users, to be sent via the SENDFILE command.
RECEIVE	Read onto disk, a file or note that is in your virtual reader.
SENDFILE	Send files or notes to one or more users on your system or a system that is attached to yours via Remote Spooling Communications Subsystem (RSCS) by issuing the command or by using a menu (display terminal only).
TELL	Send a message to one or more computer users who are logged on to your computer or to one attached to yours via RSCS.

What is a Names File?

A names file is a collection of information about other users with whom you communicate. Having a names file makes it easier for you to communicate with others because you can assign nicknames to them. An "entry" in a names file contains all of the information associated with a a particular nickname that you enter on one menu. You can also create an entry for a list of names, where the nickname would refer to the whole list.

When you create nicknames, be sure to avoid using one of the special characters defined for your virtual machine. Otherwise, unpredictable results may occur for commands using the nickname. For example, your system may be set to use '#' for the logical line end symbol (LINEND). Therefore, you should not set up nicknames to include the '#' symbol.

Following is a list of these special symbols and their default values:

Character	Symbol
Logical character delete (CHARDEL)	@
Logical line end (LINEND)	#
Logical line delete (LINEDEL)	¢
Logical escape character (ESCAPE)	"

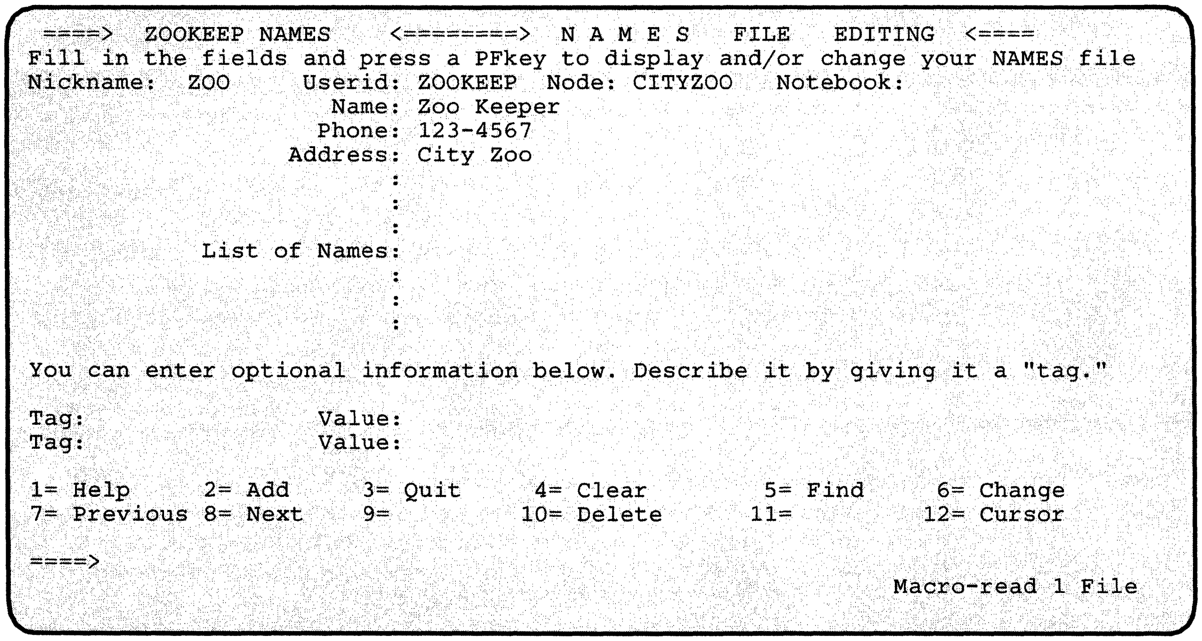
For further information on these special characters, consult the "Logical Line Editing Symbols" section of Appendix A.

Creating a Names File

The first entry in your names file should be for yourself. The information will be used for note headings, which are discussed later on. To display the names screen, enter:

```
names
```

When you enter NAMES, your userid appears (automatically) in the first line. The following is an entry in the file "ZOOKEEP NAMES":



```
 ====>  ZOOKEEP NAMES     <========>  N A M E S   FILE   EDITING  <====
Fill in the fields and press a PFkey to display and/or change your NAMES file
Nickname:  ZOO      Userid: ZOOKEEP  Node: CITYZOO   Notebook:
                      Name: Zoo Keeper
                     Phone: 123-4567
                   Address: City Zoo
                          :
                          :
                          :
             List of Names:
                          :
                          :
                          :

You can enter optional information below. Describe it by giving it a "tag."

Tag:                Value:
Tag:                Value:

1= Help     2= Add     3= Quit     4= Clear       5= Find      6= Change
7= Previous 8= Next    9=         10= Delete     11=          12= Cursor

====>
                                                        Macro-read 1 File
```

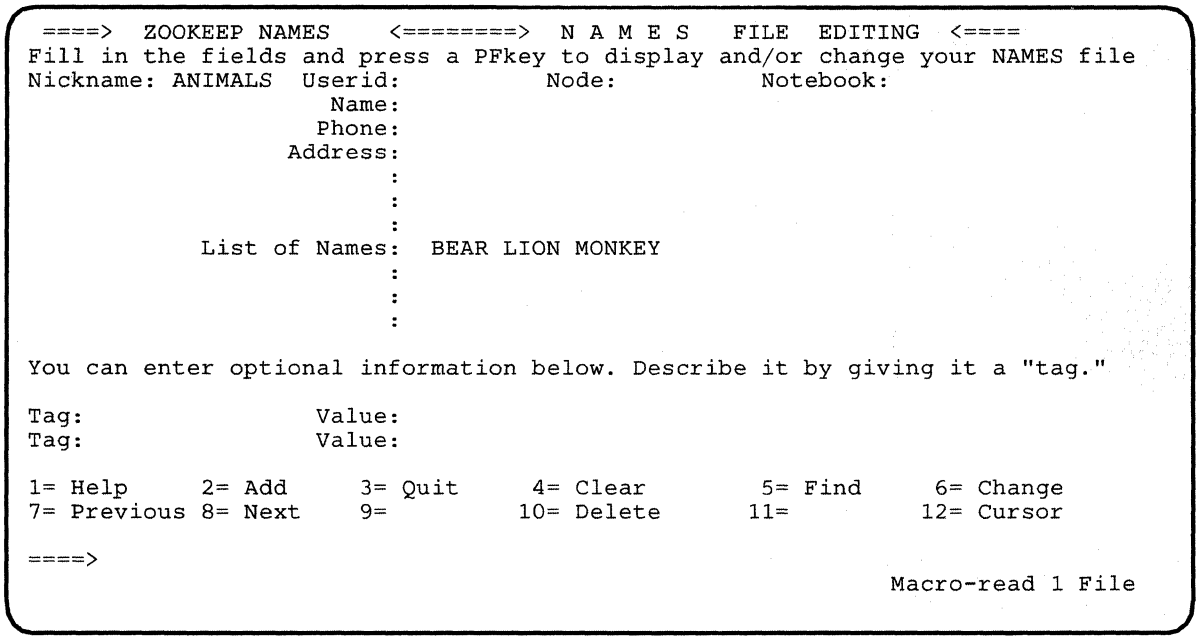
Figure 23. Sample NAMES Screen

Entering a List of names

The list of names is something like a distribution list. If you send notes, files, or messages to groups of people, you can create an entry in your names file for each group. In this case, the nickname represents the name that you want to call this list. You can specify the names of the people in the list in the following ways:

- As a nickname of an entry in the names file
- As a userid of a user who shares your computer
- In the form “userid AT node”.

Each time you send a note, a file, or a message to the nickname specified, it will go to everyone on this list. The following menu shows an entry for a list of names. Each name in the list is the nickname of an entry in the names file.

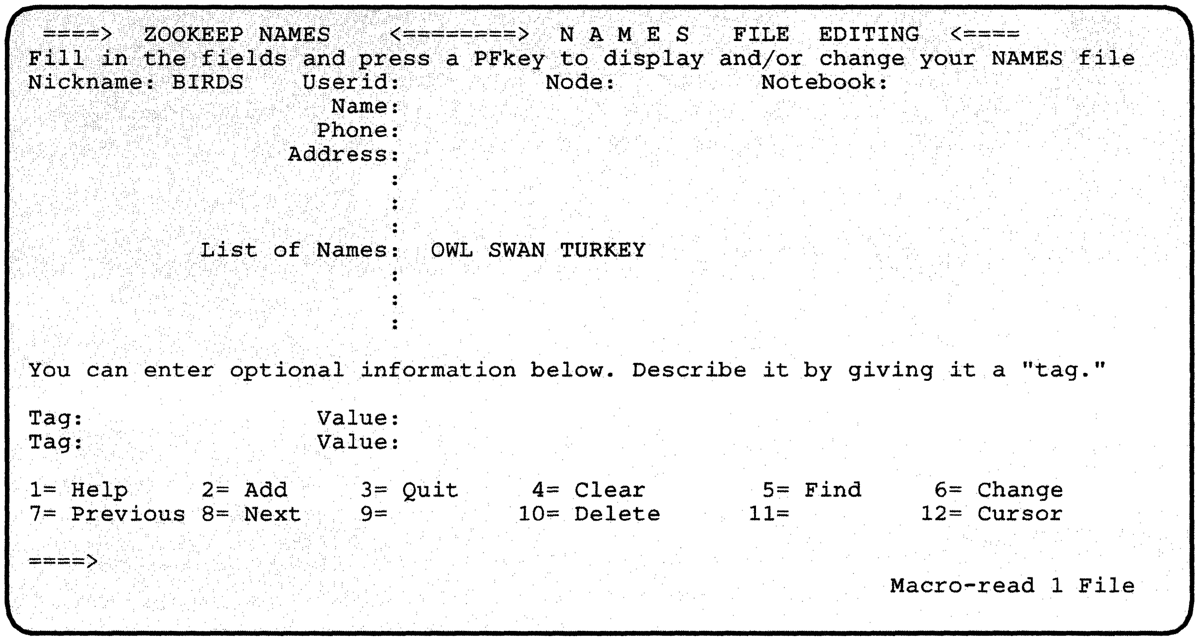


```
 ====>  ZOOKEEP NAMES    <========>  N A M E S   FILE  EDITING  <====
Fill in the fields and press a PFkey to display and/or change your NAMES file
Nickname: ANIMALS  Userid:          Node:          Notebook:
                     Name:
                    Phone:
                  Address:
                         :
                         :
                         :
            List of Names:  BEAR LION MONKEY
                         :
                         :
                         :

You can enter optional information below. Describe it by giving it a "tag."

Tag:                 Value:
Tag:                 Value:

1= Help      2= Add     3= Quit     4= Clear       5= Find      6= Change
7= Previous  8= Next    9=         10= Delete     11=          12= Cursor

====>
                                                          Macro-read 1 File
```

Figure 24. Sample Entry for a List of Names

Entering Chained Lists of Names

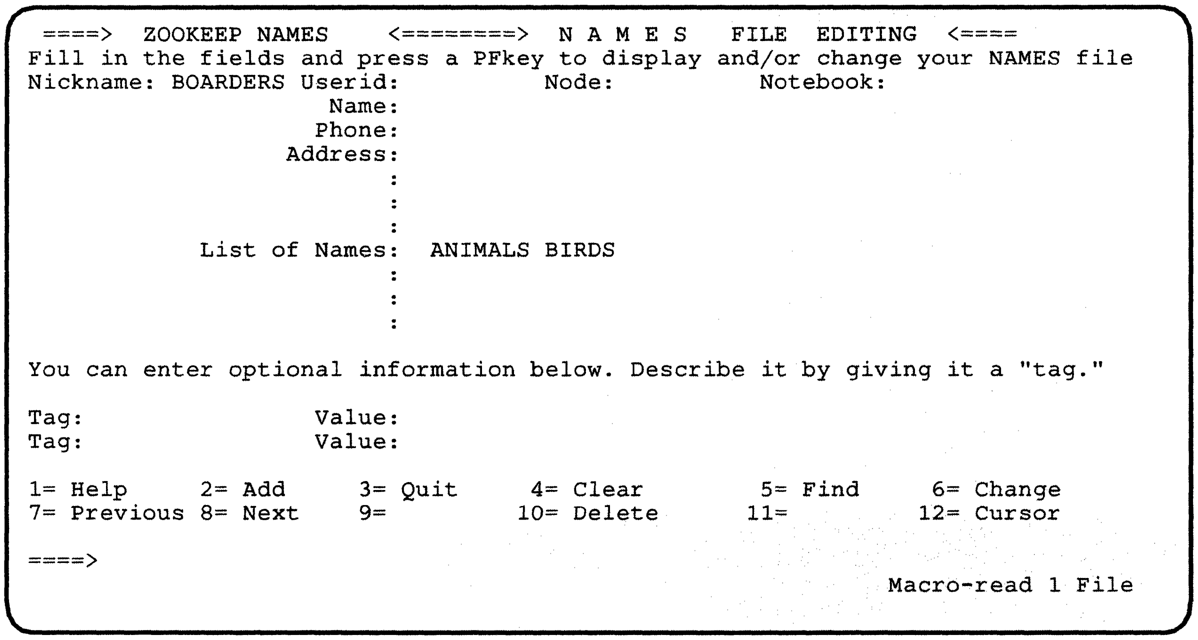
Use chained Lists of Names, to allow many users to be included in the List of Names tag. For this example, there is an entry in the ZOOKEEP NAMES file called BIRDS containing a List of Names as shown in Figure 25.



```
 ====>  ZOOKEEP NAMES     <========>  N A M E S   FILE  EDITING  <====
Fill in the fields and press a PFkey to display and/or change your NAMES file
Nickname: BIRDS    Userid:          Node:          Notebook:
                     Name:
                    Phone:
                  Address:
                         :
                         :
                         :
            List of Names:  OWL SWAN TURKEY
                         :
                         :
                         :

You can enter optional information below. Describe it by giving it a "tag."

Tag:                Value:
Tag:                Value:

1= Help     2= Add     3= Quit     4= Clear        5= Find      6= Change
7= Previous 8= Next    9=         10= Delete      11=          12= Cursor

====>
                                                          Macro-read 1 File
```

Figure 25. Another Sample Entry for a List of Names

Figure 26 shows how BIRDS and ANIMALS can be represented by two nicknames. Each name in the list is the nickname of an entry in the names file.



```
 ====>  ZOOKEEP NAMES    <========>  N A M E S   FILE  EDITING  <====
Fill in the fields and press a PFkey to display and/or change your NAMES file
Nickname: BOARDERS Userid:          Node:          Notebook:
                     Name:
                    Phone:
                  Address:
                         :
                         :
                         :
            List of Names:  ANIMALS BIRDS
                         :
                         :
                         :

You can enter optional information below. Describe it by giving it a "tag."

Tag:                Value:
Tag:                Value:

1= Help     2= Add     3= Quit     4= Clear       5= Find      6= Change
7= Previous 8= Next    9=         10= Delete     11=          12= Cursor

====>
                                                          Macro-read 1 File
```

Figure 26. Sample Entry for a Chained List of Names

If a note is sent to BOARDERS, the following receive the note:

TO NICKNAME	CHAINED LIST OF NAMES	ACTUAL RECIPIENTS
BOARDERS	ANIMALS	BEAR
		LION
		MONKEY
	BIRDS	OWL
		SWAN
		TURKEY

The following represents the ZOOKEEP NAMES file:

```
:nick.ZOO       :userid.ZOOKEEP  :node.CITYZOO
                :name.Zoo Keeper                  :phone.123-4567
                :addr.City Zoo
:nick.BEAR      :userid.GRIZZLY  :node.DEN
                :name.I. M. Grizzley              :phone.123-4567
                :addr.Den;City Zoo
:nick.LION      :userid.COWARD   :node.DEN
                :name.I.M.A. COWARD               :phone.123-4567
                :addr.Lion Den;City Zoo
:nick.MONKEY    :userid.MONKEY   :node.TREE
                :name.T.O.P. Banana               :phone.123-4567
                :addr.Banana Tree;City Zoo
:nick.ANIMALS
                :list.BEAR LION MONKEY
:nick.OWL       :userid.OWL      :node.TREE
                :name.I. M. Wise                  :phone.123-4567
                :addr.Big Tree;City Zoo
:nick.SWAN      :userid.SWAN     :node.SWANLAKE
                :name.Grace Full                  :phone.123-4567
                :addr.Swan Lake;City Zoo
:nick.TURKEY    :userid.TURKEY   :node.COTTAGE
                :name.T.TURKEY                    :phone.123-4567
                :addr.Turkey Coop;City Zoo
:nick.BIRDS
                :list.OWL SWAN TURKEY
:nick.BOARDERS
                :list.ANIMALS BIRDS
```

Figure 27. Sample 'userid NAMES' File

Sending Messages

You can send messages to one or more users on your computer or on other computers that are connected to yours via the Remote Spooling Communications Subsystem (RSCS) network. The users must be logged on to receive your message. Because the TELL command references your names file, you are able to use nicknames. For example:

```
tell bear There is honey for dessert!
```

If Bear is logged on, he sees the following message on his screen:

```
MSG FROM ZOOKEEP :  There is honey for dessert!
```

You can send a message to a list of people when you have a nickname for the list. For example:

```
tell animals Good Morning!
```

sends the message to the list of names for the nickname ANIMAL (BEAR LION, and MONKEY). They must be logged on to receive your message.

You can also use the CP MESSAGE command to send a message to a user or to the primary system operator. The recipient must be logged on to receive your message. Use the CP QUERY userid command to see if another user is logged on. The CP MESSAGE command does not use your names file. An example of the CP MESSAGE command using the abbreviation (MSG) is:

```
msg jonescj we are leaving now.
```

Receiving Messages

During the course of a terminal session, you can receive many kinds of messages from VM/SP, from the system operator, from other users, or from your own programs. You can decide whether or not you want to receive these messages. For example, if you use the command:

```
set msg off
```

You will not receive any messages sent by the TELL command or the CP MESSAGE command; if another virtual machine user tries to send you a message, he receives the message:

```
userid not receiving; MSG OFF
```

If your virtual machine handles special messages and you do not want to receive special messages at this time, you can issue:

```
set smsg off
```

You will not receive any special messages sent by the CP SMSG command; if another virtual machine user attempts to do so, he receives a message:

```
userid not receiving; SMSG OFF
```

Similarly, to prevent warning messages (which usually come from the system operator) from coming to you, you can use:

```
set wng off
```

However, you would only do this in cases where you were typing some output at your terminal and did not want the copy ruined.

VM/SP issues error messages whenever you issue a command incorrectly or if a command or program fails. These messages have a long form, consisting of the error message code and number, followed by text describing the error. If you wish to receive only the text portion of messages with severity codes I, E, and W (for informational, error, and warning, respectively), you can issue the command:

```
set emsg text
```

If you want to receive only the message code and number (from which you can locate an explanation of the error in *VM/SP System Messages and Codes*), you specify:

```
set emsg code
```

You can also cancel error messages completely:

```
set emsg off
```

To restore the EMSG setting to its default, which is the message text, enter:

```
set emsg text
```

Some CP commands issue informational messages telling you that CP has performed a particular function. You can prevent the reception of these messages with the command:

```
set imsg off
```

or restore the default by issuing:

```
set imsg on
```

The setting of EMSG applies to CMS commands as well as to CP commands.

You can also control the format of the CMS ready message. If you enter:

```
set rdymsg smsg
```

you receive only the "Ready;" or shortened form of the ready message after the completion of CMS commands. If you are not receiving error messages (as described above) and an error occurs, the return code from the command still appears in parentheses following the "Ready".

Full-screen CMS also provides several commands to control whether or not you receive messages from the system, other users, or your programs during your terminal session. In full-screen CMS, the MESSAGE window is set to pop when you receive a message. In addition, the terminal alarm will sound, and the message class indicator will indicate that you have received a message. If you do not want the window to pop automatically, issue the command SET WINDOW MESSAGE NOPOP. After you have issued this command, when you receive a message, the terminal alarm will still sound, and the message class indicator in the lower left corner of your screen will indicate that you have received a message, but the window will not pop automatically. When you are ready to view the message, you can press CMSPF 2 to pop the MESSAGE window.

Note: If you try to pop the MESSAGE window before any messages have been received, the window will not be shown. Since the window is variable in size, it will only be shown when it contains at least one message.

The default settings for the terminal alarm and message class indicator are for the alarm to sound and the indicator to be updated. However, you can change these defaults by using the ALARM/NOALARM and NOTIFY/NONOTIFY options of the ROUTE command. For example, you would use the NOALARM option if you did not want the terminal alarm to sound when you receive a message.

If you set full-screen CMS on, you can take advantage of several choices for receiving and logging messages. By default, messages and warnings will be logged for you into the files MESSAGE LOGFILE and WARNING LOGFILE, respectively. If you wish, you can use the SET LOGFILE command to log your CMS output and other information into separate files which you can later XEDIT or print.

You can also use the TYPE/NOTYPE options of the SET VSCREEN command to control how your virtual screens are updated. The TYPE option, which is the default, means that data is moved to the virtual screen when the virtual screen is updated. If you simply wanted to log messages into the file MESSAGE LOGFILE, without updating the MESSAGE virtual screen or viewing the MESSAGE window, you could specify SET VSCREEN MESSAGE NOTYPE.

For further information on the commands discussed in this section, refer to the *VM/SP CMS Command Reference.*

Sending Notes and Files

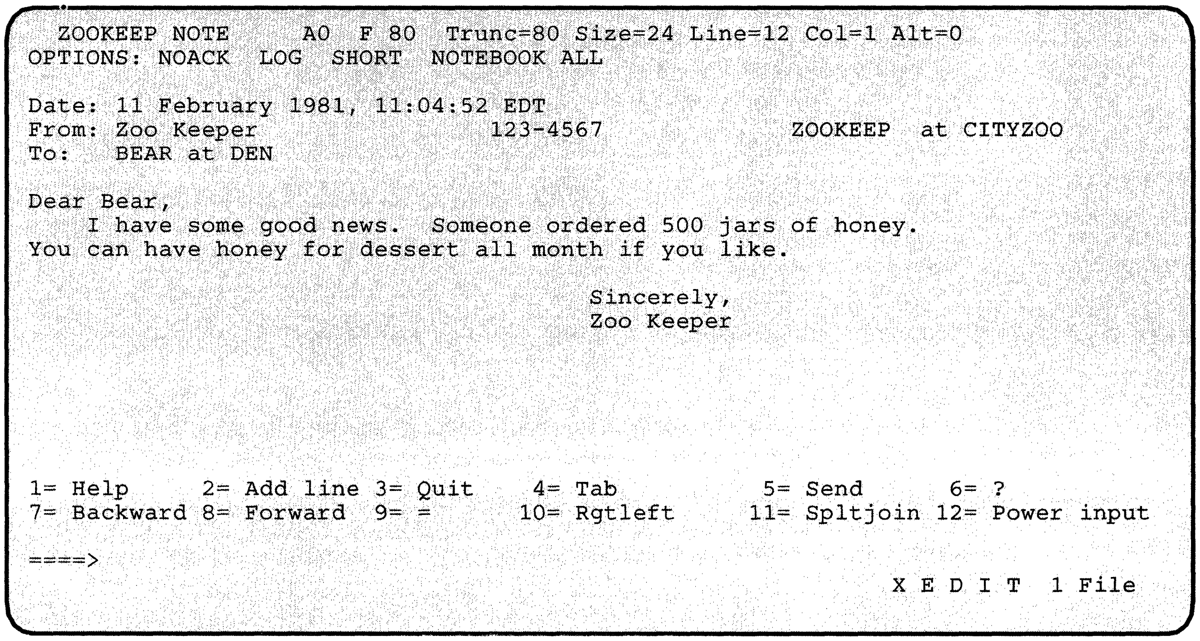
When you have short communication, like a letter, use the NOTE command to prepare a "note" to send to one or more users on your computer or on other computers that are connected to yours via the Remote Spooling Communications Subsystem (RSCS) network. The person to whom you are sending the note need not be logged on. The NOTE command references your "userid NAMES" file, so you can use nicknames when you create your notes.

Composing Notes

Entering NOTE name puts you in XEDIT mode. Enter the INPUT subcommand on the command line and type the body of the note in the space provided. Press the PF2 key to add lines if you need more space. For example:

```
note bear
```

results in a screen as in Figure 28 where the body of the note was entered in the space provided.



```
  ZOOKEEP NOTE      A0  F 80  Trunc=80 Size=24 Line=12 Col=1 Alt=0
OPTIONS: NOACK  LOG  SHORT  NOTEBOOK ALL

Date: 11 February 1981, 11:04:52 EDT
From: Zoo Keeper                123-4567             ZOOKEEP  at CITYZOO
To:   BEAR at DEN

Dear Bear,
    I have some good news.  Someone ordered 500 jars of honey.
You can have honey for dessert all month if you like.

                                       Sincerely,
                                       Zoo Keeper




1= Help     2= Add line 3= Quit    4= Tab         5= Send      6= ?
7= Backward 8= Forward  9= =      10= Rgtleft    11= Spltjoin 12= Power input

====>
                                                        X E D I T  1 File
```

Figure 28. Sample Note with Short Headings

Sending a Note

To send the note, you can do *one* of the following:

- Press the PF5 key.
- Enter on the command line one of the following:
  - SENDFILE (NOTE
  - SENDFILE (NOTE OLD

  Note: Use the OLD option when recipients do not have the RECEIVE command available to them so that the file can be DISK LOADed.
- Enter SEND, which is a synonym for SENDFILE (NOTE.

Continuing a Note

If you want to save a note and finish it later, enter FILE on the command line. The note will *not* be sent. It is saved on your disk as "userid NOTE A0." To continue the note later on, issue the NOTE command *with no parameters*.

Keeping a Copy of Notes

Each time that you send a note, a copy is kept in a file called ALL NOTEBOOK. A note is saved by appending it to the NOTEBOOK file. Notes are separated by a line of 73 equal signs (=). If you want to keep separate notebooks for certain correspondence, you can:

- Specify a notebook filename with the NOTE command. For example, to save a note in ANIMALS NOTEBOOK, enter:

  ```
  note bear (notebook animals
  ```

- Specify a notebook filename on an entry in your "userid NAMES" file.

- Set up a default notebook filename with the DEFAULTS command. Notes will go into this notebook unless you have specified a notebook filename with the NOTE command or if you have specified a notebook filename on the recipient's entry in your names file. For example to set up the default to save notes in ANIMALS NOTEBOOK, enter:

  ```
  defaults note notebook animals
  ```

See the *VM/SP CMS Command Reference* for information about the DEFAULTS command.

Sending Files

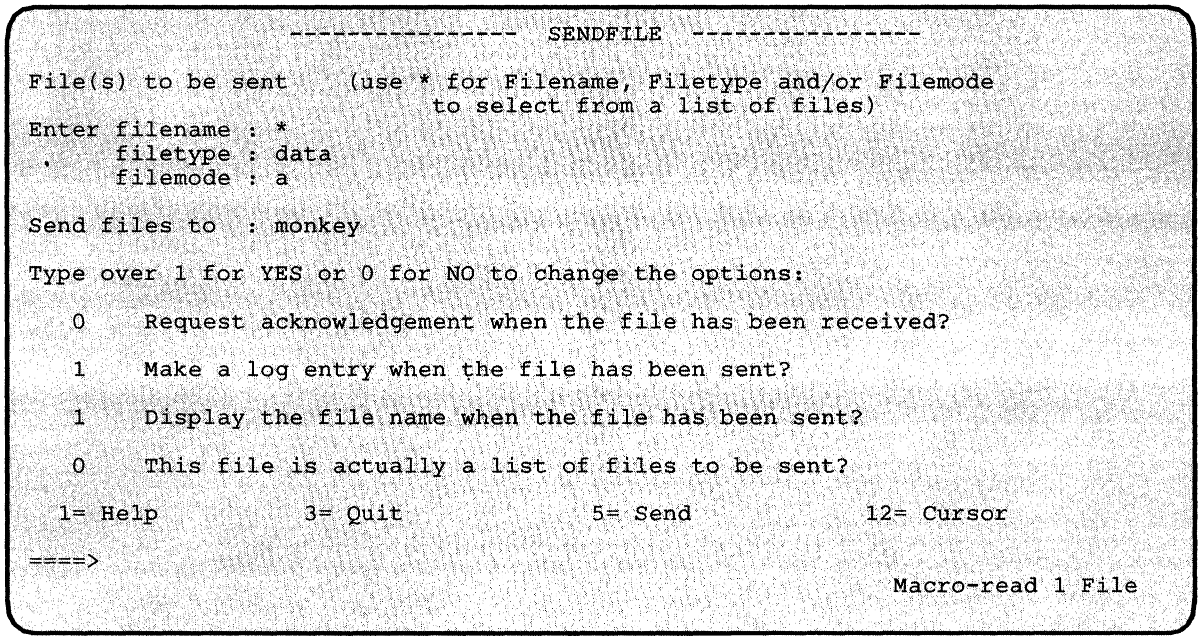
Use the SENDFILE command to send files and notes to one or more computer users on your computer or on other computers that are connected to yours via the Remote Spooling Communications Subsystem (RSCS) network. Since SENDFILE is one of the commands that references your names file, you can use nicknames to identify the recipients. The nickname is automatically converted into node and userid.

If you know the name of the file that you want to send, you can just enter the file identification and nickname following the SENDFILE command. For example:

```
sendfile banana split a to monkey
```

Otherwise, if you cannot remember the name of the file or if you have many files to send, enter SENDFILE without operands. When you enter the SENDFILE command (or the abbreviation SF), a special screen is displayed.

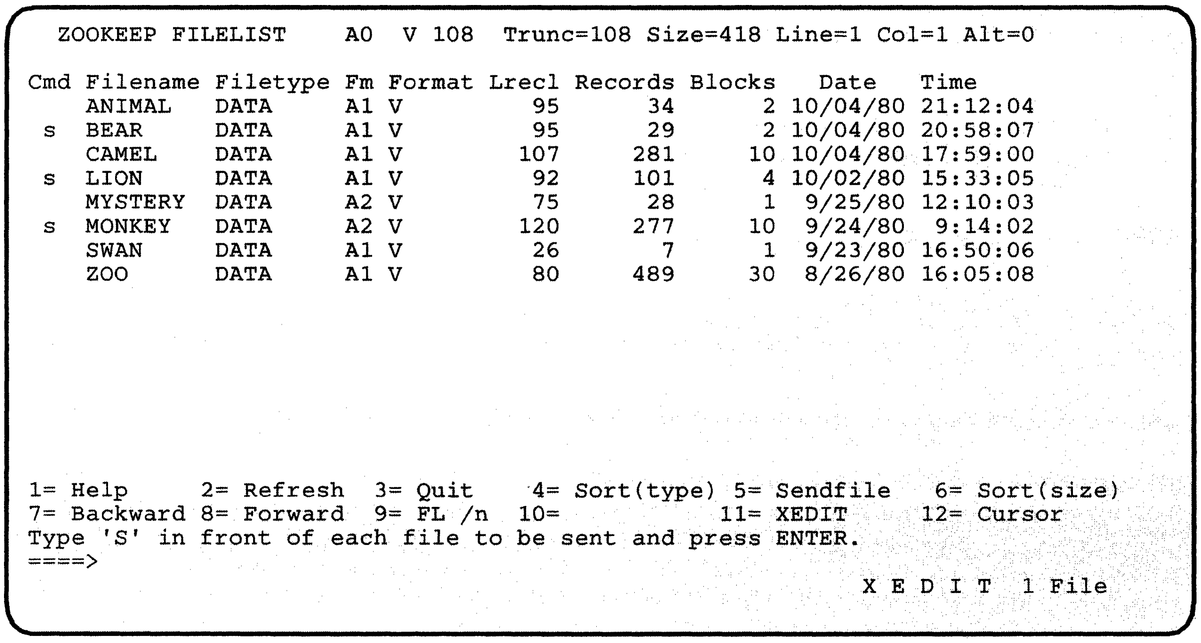
The following is a sample SENDFILE menu:



```
                    ---------------- SENDFILE ----------------

File(s) to be sent    (use * for Filename, Filetype and/or Filemode
                            to select from a list of files)
Enter filename : *
      filetype : data
      filemode : a

Send files to  : monkey

Type over 1 for YES or 0 for NO to change the options:

   0    Request acknowledgement when the file has been received?

   1    Make a log entry when the file has been sent?

   1    Display the file name when the file has been sent?

   0    This file is actually a list of files to be sent?

  1= Help          3= Quit             5= Send          12= Cursor

====>
                                                      Macro-read 1 File
```

Figure 29. Sample SENDFILE Menu

In Figure 29, the sender entered an asterisk for filename, 'data' for filetype, and 'a' for filemode. The name of the recipient (MONKEY) is also entered on the screen. When PF5 (or the ENTER key) is pressed, a special FILELIST screen is displayed. The files to be sent can be selected from this screen (shown in Figure 30 on page 167).



```
  ZOOKEEP FILELIST    A0  V 108  Trunc=108 Size=418 Line=1 Col=1 Alt=0

Cmd Filename Filetype Fm Format Lrecl Records Blocks   Date    Time
    ANIMAL   DATA     A1 V         95      34      2 10/04/80 21:12:04
 s  BEAR     DATA     A1 V         95      29      2 10/04/80 20:58:07
    CAMEL    DATA     A1 V        107     281     10 10/04/80 17:59:00
 s  LION     DATA     A1 V         92     101      4 10/02/80 15:33:05
    MYSTERY  DATA     A2 V         75      28      1  9/25/80 12:10:03
 s  MONKEY   DATA     A2 V        120     277     10  9/24/80  9:14:02
    SWAN     DATA     A1 V         26       7      1  9/23/80 16:50:06
    ZOO      DATA     A1 V         80     489     30  8/26/80 16:05:08








1= Help     2= Refresh  3= Quit    4= Sort(type) 5= Sendfile   6= Sort(size)
7= Backward 8= Forward  9= FL /n  10=           11= XEDIT     12= Cursor
Type 'S' in front of each file to be sent and press ENTER.
====>
                                                      X E D I T  1 File
```

Figure 30. Sample FILELIST Screen Invoked from SENDFILE

To send one or more of these files, you can type a letter ‘s’ in front of the filename of each file you want sent (see above) and then press the ENTER key. You can also position the cursor on the line describing the file you want to send, and then press the PF5 key.

Sending One File

To send only one file:

1. Enter SENDFILE (or its abbreviation SF).
2. On the SENDFILE menu, enter the filename, filetype and filemode in the spaces provided. If the filemode is A, you can leave filemode blank.
3. Enter the names of the recipient(s). Remember that you can use nicknames.
4. Select the 0 or 1 options.
5. Press either PF5 or the ENTER key to send the file. Pressing:
   - PF5 sends the file and exits from the menu.
   - The ENTER key sends the file and keeps the menu.

8

Receiving Notes and Files

After you logon you might see a message notifying you that you have files in your reader. For example:

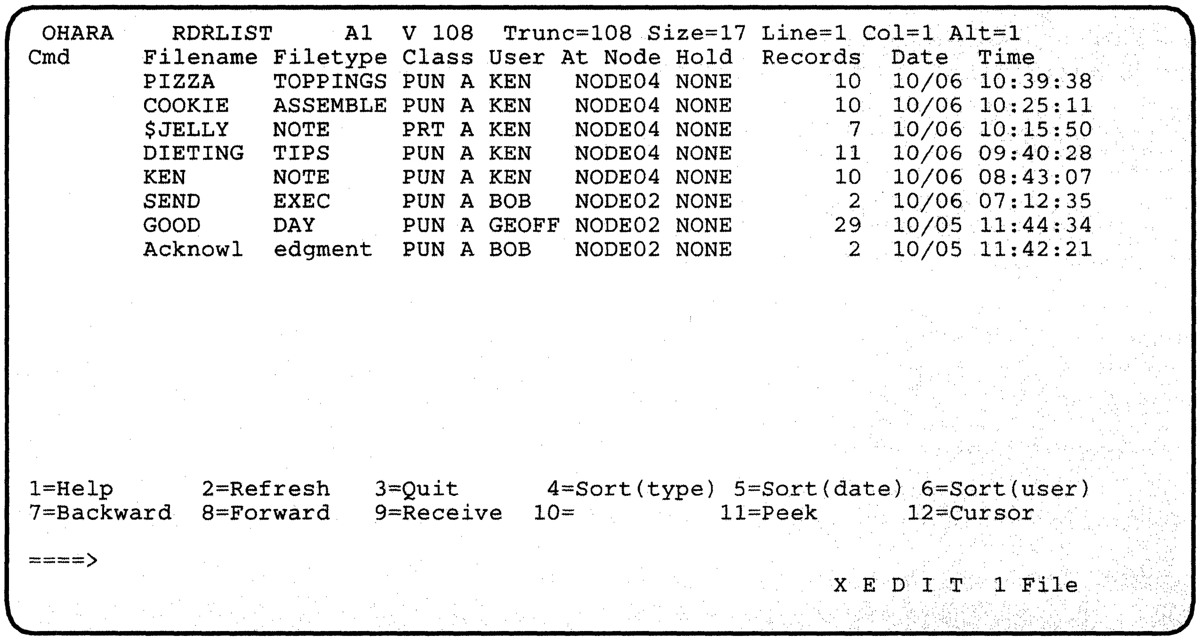
```
FILES:  004 RDR, NO PRT, NO PUN
```

During your terminal session if you want to find out if you have files in your virtual reader you can enter any of the following commands. The table below shows the command, provides a description, and the system response if there are no files in your reader.

RDRLIST	Displays information about the files in your virtual reader with the ability to issue commands from the list.	No files in your reader. Ready(00028);
CP QUERY RDR ALL	Lists your reader files (if any) and their characteristics.	NO RDR FILES Ready;
CP QUERY FILES	Displays the number of spool files in your virtual machine.	FILES: NO RDR, NO PRT, NO PUN Ready;

Using the CMS RDRLIST Command

The command CMS RDRLIST displays information about the files in your reader. For example:



```
 OHARA    RDRLIST      A1  V 108  Trunc=108 Size=17 Line=1 Col=1 Alt=1
Cmd      Filename Filetype Class User At Node Hold  Records  Date  Time
         PIZZA    TOPPINGS PUN A KEN    NODE04 NONE       10  10/06 10:39:38
         COOKIE   ASSEMBLE PUN A KEN    NODE04 NONE       10  10/06 10:25:11
         $JELLY   NOTE     PRT A KEN    NODE04 NONE        7  10/06 10:15:50
         DIETING  TIPS     PUN A KEN    NODE04 NONE       11  10/06 09:40:28
         KEN      NOTE     PUN A KEN    NODE04 NONE       10  10/06 08:43:07
         SEND     EXEC     PUN A BOB    NODE02 NONE        2  10/06 07:12:35
         GOOD     DAY      PUN A GEOFF  NODE02 NONE       29  10/05 11:44:34
         Acknowl  edgment  PUN A BOB    NODE02 NONE        2  10/05 11:42:21







1=Help       2=Refresh   3=Quit      4=Sort(type) 5=Sort(date) 6=Sort(user)
7=Backward   8=Forward   9=Receive  10=          11=Peek      12=Cursor

====>
                                                   X E D I T  1 File
```

Figure 31. Sample RDRLIST Screen

Some of the commands that you can issue from the list are:

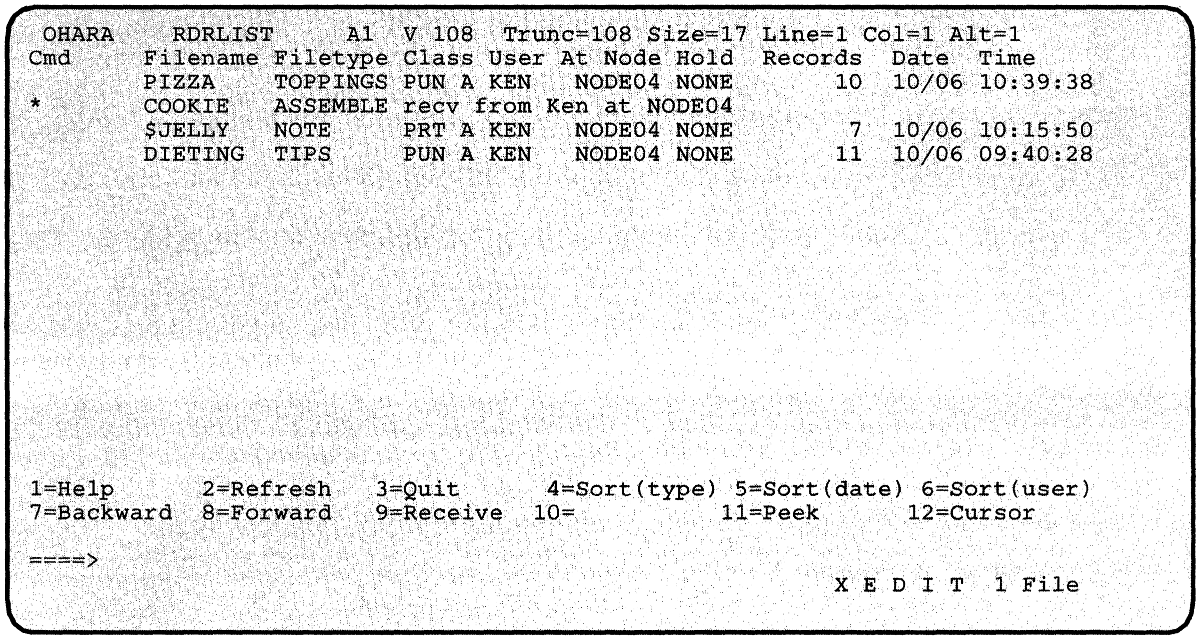
PEEK	Displays a file in your virtual reader without reading it onto disk.
RECEIVE	Reads onto disk a file or note that is in your virtual reader.
DISCARD	Purges a file displayed in the reader list.

Receiving a File

If you have issued the RDRLIST command and you want to receive a file:

1. Move the cursor to the line describing the file that you want to receive.
2. Press PF9. A notice will appear on that line, telling you that the file has been received.

For example:



```
 OHARA    RDRLIST     A1  V 108  Trunc=108 Size=17 Line=1 Col=1 Alt=1
Cmd     Filename Filetype Class User At Node Hold  Records  Date  Time
        PIZZA    TOPPINGS PUN A KEN   NODE04 NONE       10  10/06 10:39:38
*       COOKIE   ASSEMBLE recv from Ken at NODE04
        $JELLY   NOTE     PRT A KEN   NODE04 NONE        7  10/06 10:15:50
        DIETING  TIPS     PUN A KEN   NODE04 NONE       11  10/06 09:40:28









1=Help      2=Refresh   3=Quit      4=Sort(type) 5=Sort(date) 6=Sort(user)
7=Backward  8=Forward   9=Receive  10=          11=Peek      12=Cursor

====>
                                                     X E D I T  1 File
```

Figure 32. Sample RDRLIST Screen after Receiving a File

If the file in your reader has the same name as a file that is already on your disk, after you press PF9, you will receive the following message on your RDRLIST screen:

```
File fn ft fm already exists; specify REPLACE option
```

If you want to replace the file on your disk, enter the following in the "Cmd" space next to the file that you want to receive:

```
receive / (replace
```

clear the remainder of the line, and press the ENTER key. The file on your disk will be replaced.

If you want to keep the file on your disk, you can either:

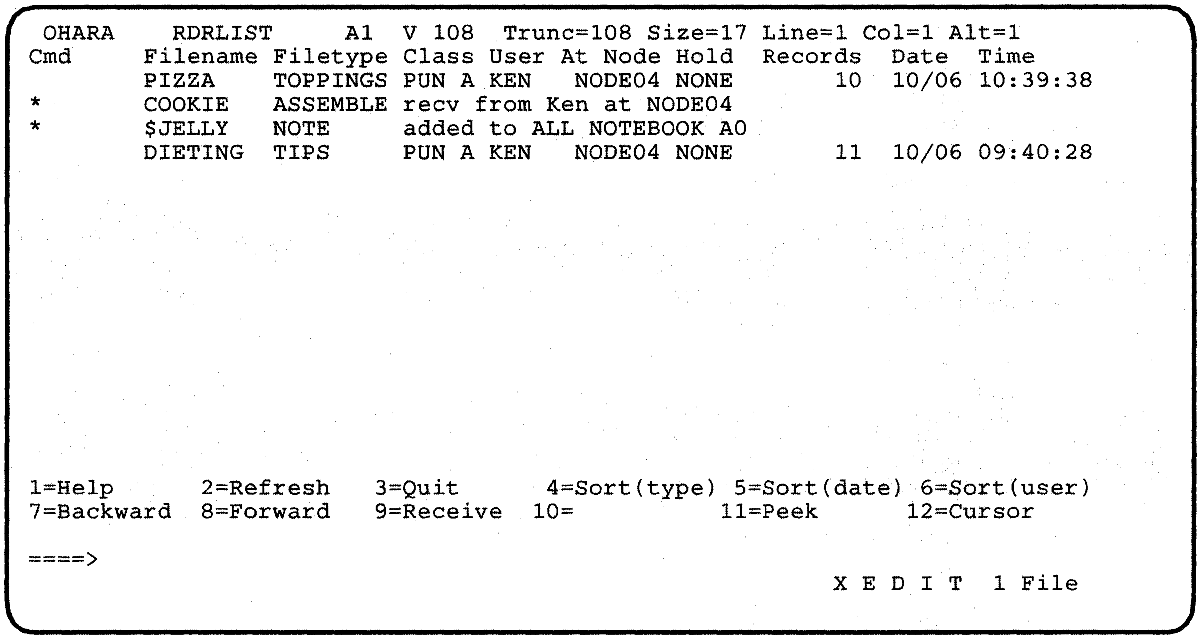
- Rename the file on your disk (use the CMS RENAME command), or
- Give the file to be read in a new name. For example, enter the following in the "Cmd" space and press the ENTER key:

  ```
  receive / banana split
  ```

Receiving a Note

You can receive notes in the same way that you receive other types of files. A note will have a filetype of NOTE. Just move the cursor to the line describing the note and press PF9.

The note is appended to your ALL NOTEBOOK file, unless you have a notebook specified in your names file entry for that person. For example, after receiving $JELLY NOTE, the sample screen would look like the following:



```
 OHARA    RDRLIST     A1  V 108  Trunc=108 Size=17 Line=1 Col=1 Alt=1
Cmd       Filename Filetype Class User At Node Hold  Records  Date  Time
          PIZZA    TOPPINGS PUN A KEN   NODE04 NONE       10  10/06 10:39:38
*         COOKIE   ASSEMBLE recv from Ken at NODE04
*         $JELLY   NOTE     added to ALL NOTEBOOK A0
          DIETING  TIPS     PUN A KEN   NODE04 NONE       11  10/06 09:40:28











1=Help      2=Refresh   3=Quit      4=Sort(type) 5=Sort(date) 6=Sort(user)
7=Backward  8=Forward   9=Receive  10=          11=Peek      12=Cursor

====>
                                                   X E D I T  1 File
```

Figure 33. Sample RDRLIST Screen after Receiving a Note

Discarding a File

Use the DISCARD command to purge a file displayed in your RDRLIST file. Unlike the CP PURGE command, DISCARD allows an acknowledgment to be sent to the sender (if one was requested). The acknowledgment indicates that the file was discarded. DISCARD also makes an entry in your "userid NETLOG" file, (if the log option was in effect in the RECEIVE command) indicating that the file was discarded. To discard a file, enter DISCARD on the "Cmd" space next to the file that you want to discard and press the ENTER key.

Chapter 9. Looking at VM/SP Through Windows

Windowing allows you to manage several pieces of information on the physical screen at the same time. Through windows, you can manipulate information as you might rearrange pieces of paper on your desk top.

Windowing support consists primarily of the following:

- Windows
- Virtual Screens

Working with windows has several advantages. When you set full-screen CMS on, special windows are defined for you automatically, such as the MESSAGE window. If, for example, you receive a message while you are editing a file, you can look at it through the MESSAGE window without leaving the file you are editing. Or, you can hold the message until you have time to look at it later.

This chapter explains windowing support. It is divided into two sections: "What are Windows and Virtual Screens?" and "Using Full-Screen CMS."

"What are Windows and Virtual Screens?" defines windows and virtual screens. "Using Full-Screen CMS" discusses how CMS can have functions similar to XEDIT such as special PF keys and full-screen input for CMS commands.

What Are Windows and Virtual Screens?

A virtual screen can be thought of as a "presentation space" where data can be stored. A virtual screen (vscreen for short) simulates a physical screen but is not confined to the physical screen's size.

A window is an area on the physical screen where virtual screen data can be displayed and manipulated. A window let's you see what's in a virtual screen.

When you are looking at a window, you are actually viewing a virtual screen. Depending on the size of the window and the size of the virtual screen, you may be seeing a portion of the virtual screen or the entire virtual screen. For more information on virtual screens, refer to the DEFINE VSCREEN command in the *VM/SP CMS Command Reference.*

Because a window shows you a portion of a virtual screen, you can perform several operations against the data in a virtual screen and view the results in the window connected to the virtual screen. The characteristics of virtual screens that you can manipulate include:

- Reserved areas for information such as titles and PF key descriptions
- Color and highlighting
- Options to log data into a file.

The following diagram illustrates the relationship between the physical screen, a window, and a virtual screen.

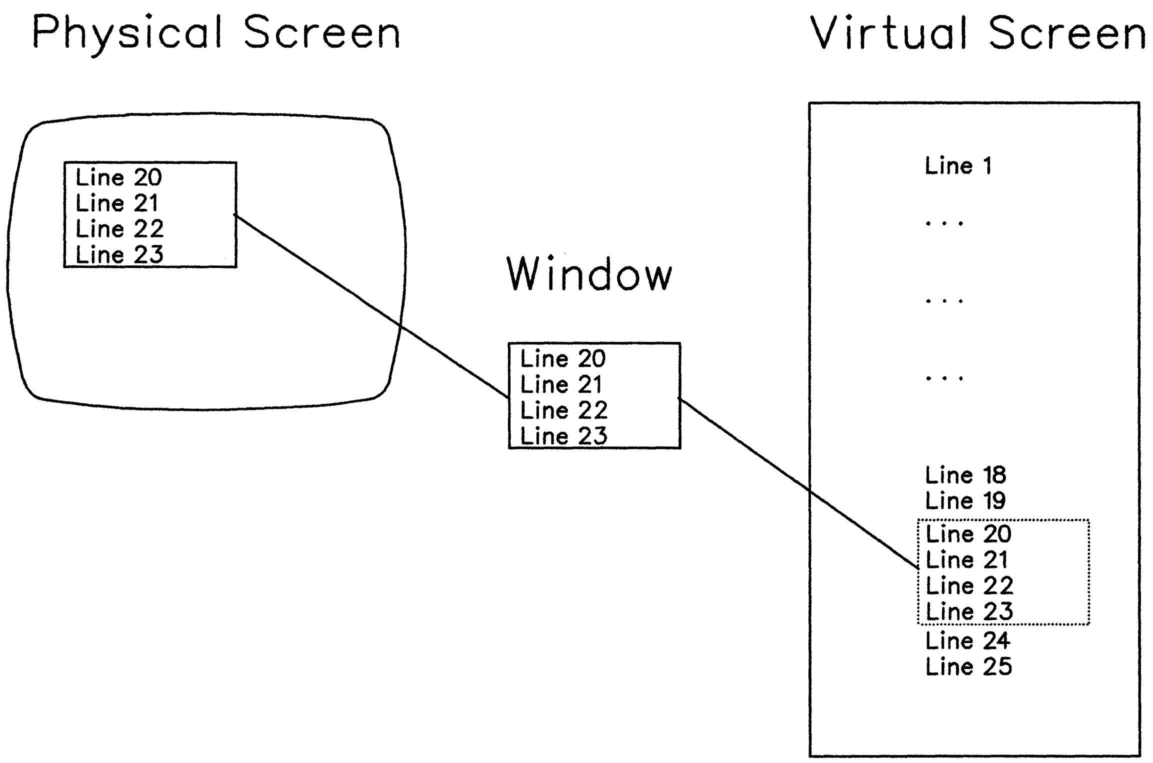


Figure 34. A Window into a Virtual Screen

When working with windows, you do not have to be concerned with the internal interactions between windows and virtual screens. However, as you become more familiar with how they work, you may find it useful to manipulate information by using the CMS commands for windows and virtual screens. Some of these commands will be discussed in this section. For further information on each command, refer to the *VM/SP CMS Command Reference.*

What's in a Window?

Windows can be positioned anywhere on the screen as long as the entire window fits on the screen. The maximum size of any window is the size of the screen. You can have many windows on the screen at once. The windows may be displayed on top of each other and may overlap.

When you manipulate data in a window, you are actually manipulating the data in a virtual screen. Virtual screen data can be viewed by scrolling the window over the virtual screen.

Windows are maintained in an ordered list. You can shuffle the order by "popping" and "dropping" windows. The CMS commands to do this are POP WINDOW and DROP WINDOW.

Default windows are those that the system defines for you when you issue SET FULLSCREEN ON. Default windows may have reserved areas at the top and bottom, where system information is shown, and a scrollable data area between the reserved areas. In addition, location information may be displayed in the upper right corner of the windows.

The following section shows a sample full-screen CMS window and provides detailed information on the parts of the window.

Using Full-Screen CMS

There are many advantages to using full-screen CMS. When you set full-screen on, you can type commands from almost anywhere on the physical screen. You can scroll forward and backward through your CMS session to see commands you entered previously and CMS responses to these commands. To re-issue any command, you do not need to re-type the entire command. Instead, scroll back until the command is displayed on your screen, position your cursor on the command, type over any letter, and press ENTER. The command will be re-executed.

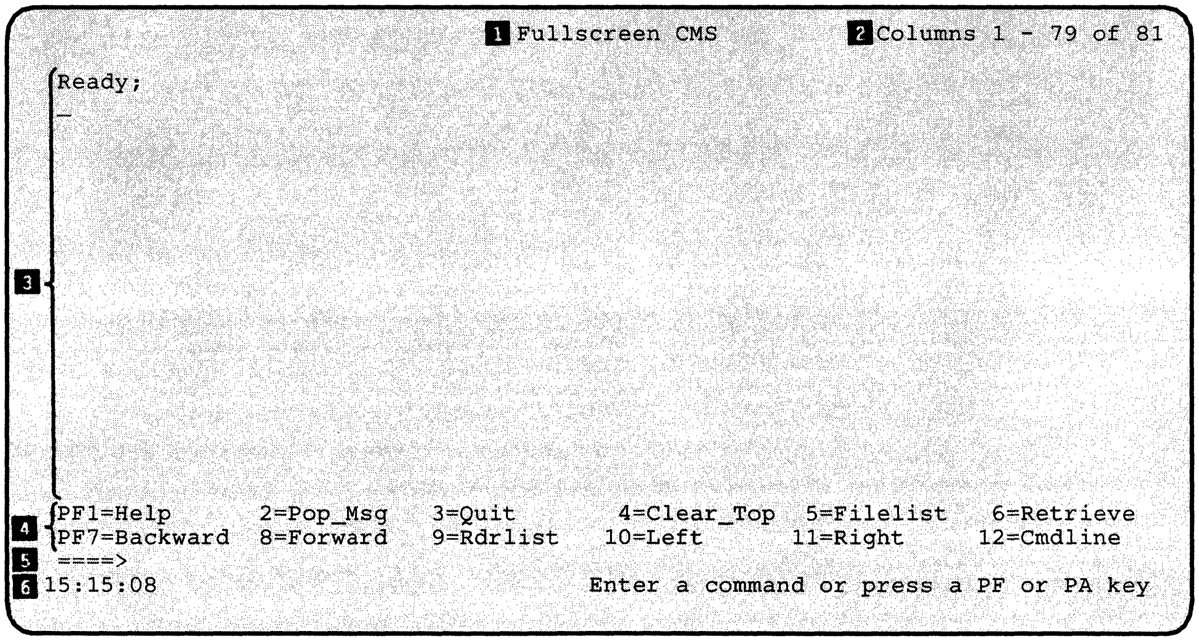
During your full-screen CMS session, messages and other output appear in windows on your physical screen and can be viewed without leaving your current work environment.

After you have logged on to the system, you can type SET FULLSCREEN ON. Or, you can put this command in your PROFILE EXEC. Once you have issued this command, CMS is in a window and can take advantage of windowing support.

If you wish to leave full-screen CMS, just issue SET FULLSCREEN OFF. You can also issue SET FULLSCREEN SUSPEND or press the CMSPF 3 key to suspend full-screen CMS. The advantage to suspending full-screen CMS is that you can later issue SET FULLSCREEN RESUME and re-enter your full-screen session where you left off. None of the default or user-defined settings for windows, virtual screens, or PF keys would be lost.

Let's try some examples in full-screen CMS. Note that the examples and screens in this chapter are based on a physical screen size of 24 lines by 80 columns.

Enter SET FULLSCREEN ON from the command line. Your screen now looks like this:



```
                                    1 Fullscreen CMS          2 Columns 1 - 79 of 81
  Ready;
  _
3





4 PF1=Help        2=Pop_Msg   3=Quit      4=Clear_Top  5=Filelist   6=Retrieve
  PF7=Backward    8=Forward   9=Rdrlist   10=Left      11=Right     12=Cmdline
5 ====>
6 15:15:08                              Enter a command or press a PF or PA key
```

Figure 35. Full-Screen CMS

Before you enter data, let's look at how your physical screen is organized.

1 Title Line	identifies full-screen CMS. This is the CMS window. Other windows that have a title line are the MESSAGE, WARNING, and NETWORK windows.
2 Location Information	shows the location of the window on the virtual screen. It appears in the upper right corner of the window when there may be additional virtual screen data to be displayed. For more information, refer to the section entitled "Location Information" on page 178.
3 Scrollable Data Area	is the area of the window where CMS output is displayed. You can enter commands anywhere in this area, and you can scroll backwards and forwards through the data displayed.

4 PF Key Definition Area displays the CMSPF keys and their functions. Each function is described with a 9-character pseudonym which represents a command. Users can change the function and pseudonym assigned to a key. For now, we will use the pre-assigned or "default" settings.

5 Command Line is the area of the window, in addition to the scrollable data area, where commands can be typed. This area is indicated by the large arrow.

6 Status Area reflects conditions or states which exist in your virtual machine.

The status area contains the following indicators:

Clock
indicates the current time in hours, minutes, and seconds.

Message Class
indicates that a message was received by the virtual machine. Each time a message is received, its message class will appear in this area. One of the following default message classes may be displayed in this area:

- Message
- Warning
- Network.

Execution State
reflects the status of the session. The next section gives a description of each response which may appear in the status area.

Status Information

During a full-screen CMS session, one of the following execution state responses will appear in the status area, which is the lower right corner of the physical screen.

Executing a command
The system is processing your command.

Enter your response in vscreen *'vname'*
The system is waiting for your reply to a request.

Note: In this message, *'vname'* will be replaced by the name of the virtual screen in which you are to enter your response.

Scroll forward for more information in vscreen *'vname'*
The system is waiting for you to scroll forward a window that is connected to the specified virtual screen.

Note: In this message, *'vname'* will be replaced by the name of a virtual screen.

When you receive this message, it means that new information is waiting to be added to the virtual screen. If you scroll a window connected to the virtual screen (or, in the instance when multiple windows on your screen are showing the virtual screen, if you scroll the window showing the virtual screen that is closest to the top of the ordered list of windows), the virtual screen will be updated with the new information. The virtual screen will also be updated if you enter one of the following commands: CLEAR WINDOW, CLEAR VSCREEN, SHOW WINDOW, or HIDE WINDOW. When a virtual screen is updated, this means that the oldest information in the virtual screen will be deleted from the top and the new information will be added to the bottom. This updating process will occur even if the window connected to the virtual screen is hidden or overlaid by other windows.

Enter a command or press a PF or PA key
The system is waiting to process your next input.

Note: If you set autoread on, "Enter a command or press a PF or PA key" will be replaced by the status notice, "Enter your response in vscreen CMS" following each command.

Whenever there is a conflict between status notices, the highest priority status notice will be displayed. In the list above, the status notices are listed in order of priority, from highest to lowest.

Location Information

This information is specified in the following ways. You will see this data in the upper right corner of your window when there may be additional virtual screen data to be displayed.

Lines x - y of z
indicates the position of the window in relation to lines of virtual screen data. In other words, it shows which lines you are seeing in relation to how many more you can view.

Note: The CMS virtual screen is filled starting at the top. Lines are added sequentially until the virtual screen is full. Once the virtual screen is full, and you continue to work, older data which you have already scrolled begins to be deleted off the top of the virtual screen; new data is added to the bottom. Lines that you have not yet scrolled are not deleted.

Since old lines are deleted when new ones are added, the lines are renumbered. For this reason, the location information may appear to remain constant as you scroll forward. However, the data will change.

Columns x - y of z
indicates the position of the window in relation to columns of virtual screen data. Depending on the placement of the window, there may be more data to the right or left of your window.

Your Default Windows and Virtual Screens

When you are in full-screen CMS, several windows and virtual screens are available to you automatically. The windows and the virtual screens to which the windows are connected are listed in the following table:

Window	Virtual Screen	Description
CMS	CMS	Displays CMS and CP output
CMSOUT	CMS	Displays CMS and CP output while in XEDIT or the productivity aids which use XEDIT (FILELIST, RDRLIST, etc.)
MESSAGE	MESSAGE	Displays user messages
NETWORK	NETWORK	Displays network messages
STATUS	STATUS	Displays status messages
WARNING	WARNING	Displays warnings
WM	WM	Provides the capability to enter windowing commands

Figure 36. Default Windows and Virtual Screens

The examples throughout this chapter show only the default windows and virtual screens. You can find more information on the characteristics of these windows and virtual screens by referring to the tables at the end of Chapter 15, "Customizing Full-Screen CMS."

The WM Window

The WM window is a special window for window manipulation. That is, for dropping, moving, or changing the size of other windows. You can choose to display the window at any time during your CMS session. There are also certain situations when the WM window will automatically be displayed on your screen. Depending on how the WM window was invoked, one of the following three messages will appear in the window:

- Active window overlaid; enter a windowing command or press a PF key
- Output displayed; enter a windowing command or press a PF key
- Enter a windowing command or press a PF key

The WM window will automatically be displayed on your screen in the following situations:

- When the entire screen is protected and the active window is overlaid. For example, this situation could occur if you maximize a window so that it fills the entire screen and covers all other windows. If the window is protected, you would not be able to enter any commands in it. In this case, when the WM window is displayed, you could return the window to the way it was prior to the maximize command by pressing WMPF 12 (which defaults to the command RESTORE WINDOW =) and then drop the WM window by pressing WMPF 3.

  When the WM window appears because the active window was overlaid, the window will display the message, "Active window overlaid; enter a windowing command or press a PF key."

- When you run an application that uses the CONSOLE macro to perform I/O and

  - The CMS virtual screen is updated, or
  - Any virtual screen (other than CMS) is updated and a pop-type window is showing it.

  In this instance, you can simply drop the WM window to return to the application.

  When the WM window appears while you are running an application using the CONSOLE macro and the above conditions are true, the window will contain the message, "Output displayed; enter a windowing command or press a PF key."

If you wish, you can also pop the WM window at any time during your CMS session. To do this, you would simply enter the command POP WINDOW WM or press the PA1 key (which in full-screen CMS defaults to the command POP WINDOW WM).

When you pop the WM window, the window will display the message, "Enter a windowing command or press a PF key."

The WM window provides you with a command line and a set of PF keys so that you can type commands to manipulate the windows which are covering up your screen. You can execute commands with the WMPF keys, or you can enter windowing commands from the command line in the WM window.

Following is a list of all the commands you can enter from the WM window:

CLEAR WINDOW
CP
DROP WINDOW
PUT SCREEN
QUERY BORDER
QUERY HIDE
QUERY LOCATION
QUERY RESERVED
QUERY SHOW
HELP
HIDE WINDOW
MAXIMIZE WINDOW
QUERY WINDOW
QUERY WMPF
RESTORE WINDOW
SCROLL
SET BORDER
SET LOCATION
MINIMIZE WINDOW
POP WINDOW
POSITION WINDOW
SET RESERVED
SET WINDOW
SET WMPF
SHOW WINDOW
SIZE WINDOW

As you can see, the WM window is very useful since it provides you with a means of entering commands when you may not have access to the CMS command line or the CMS window.

To drop the WM window, you can press WMPF 3. This returns you to the CMS window.

As an alternative to pressing WMPF 3 to drop the WM window, you can also use the CLEAR key. The CLEAR key scrolls the topmost window forward. When there is no more data to scroll, you will exit from the WM environment.

In the section entitled "Using the WM Window" on page 198, we will further discuss the WM window and practice using it.

Special Keys

When you are in full-screen CMS, you have access to the CMSPF keys. As we discussed in the section above, when the WM window is displayed on your screen, you can use the WMPF keys. The PA1 and PA2 keys also have special settings in full-screen CMS. The PA1 key pops the WM window, and the PA2 key can be used to scroll the top window forward. In addition, the CLEAR key serves the same purpose as the PA2 key.

This section explains how to use the PA and PF keys and the CLEAR key to simplify your work and provides information on the default PF key settings.

The CMSPF Keys

Looking again at your physical screen with full-screen CMS on, let's work with the CMSPF keys.

Each key is given a pseudonym, which represents a command, as shown in the following table:

CMSPF Key	Pseudonym	Command
CMSPF 1	Help	ECHO HELP
CMSPF 2	Pop_Msg	NOECHO #WM POP WINDOW MESSAGE *
CMSPF 3	Quit	NOECHO SET FULLSCREEN SUSPEND
CMSPF 4	Clear_Top	NOECHO #WM CLEAR WINDOW =
CMSPF 5	Filelist	ECHO EXEC FILELIST
CMSPF 6	Retrieve	RETRIEVE
CMSPF 7	Backward	NOECHO #WM SCROLL BACKWARD CMS 1
CMSPF 8	Forward	NOECHO #WM SCROLL FORWARD CMS 1
CMSPF 9	Rdrlist	ECHO EXEC RDRLIST
CMSPF 10	Left	NOECHO #WM SCROLL LEFT CMS 10
CMSPF 11	Right	NOECHO #WM SCROLL RIGHT CMS 10
CMSPF 12	Cmdline	NOECHO CURSOR VSCREEN CMS -2 8 (RESERVED

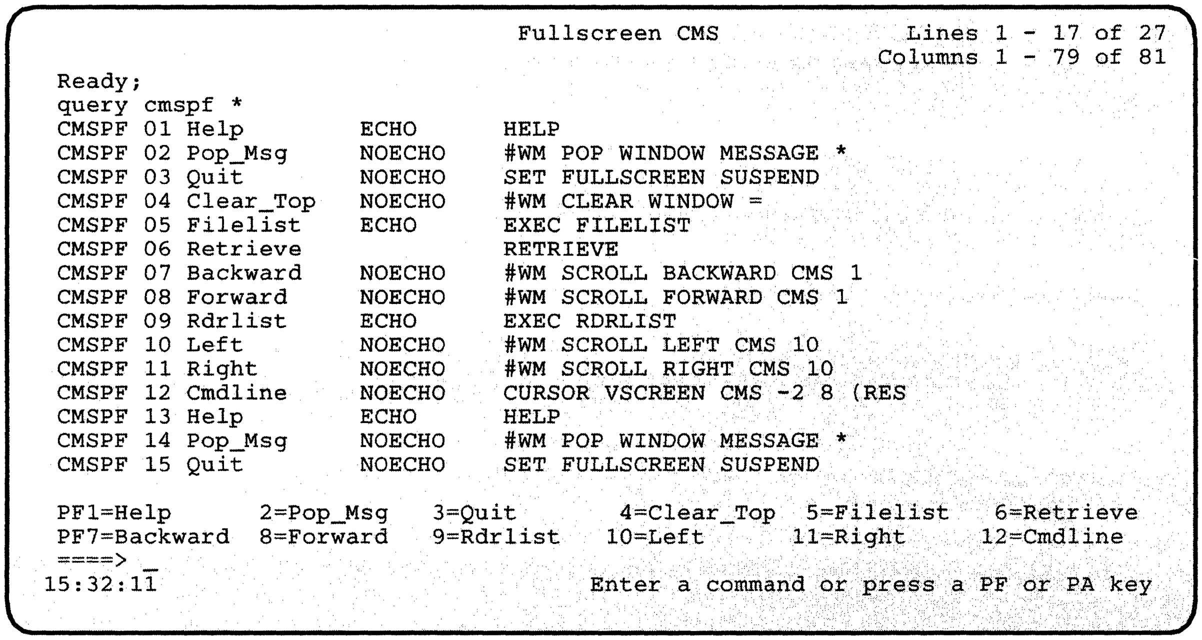
Figure 37. CMSPF Key Settings

For terminals equipped with 24 PF keys, PF keys 13 through 24 have the same values as PF keys 1 through 12, respectively. If you want to see the complete list of commands assigned to all the CMSPF keys, you can type:

```
query cmspf *
```

and press the ENTER key. You may have to scroll forward to see all the settings.

The CMS commands assigned to the PF keys will appear like this:



```
                                 Fullscreen CMS              Lines 1 - 17 of 27
                                                           Columns 1 - 79 of 81
   Ready;
   query cmspf *
   CMSPF 01 Help        ECHO       HELP
   CMSPF 02 Pop_Msg     NOECHO     #WM POP WINDOW MESSAGE *
   CMSPF 03 Quit        NOECHO     SET FULLSCREEN SUSPEND
   CMSPF 04 Clear_Top   NOECHO     #WM CLEAR WINDOW =
   CMSPF 05 Filelist    ECHO       EXEC FILELIST
   CMSPF 06 Retrieve               RETRIEVE
   CMSPF 07 Backward    NOECHO     #WM SCROLL BACKWARD CMS 1
   CMSPF 08 Forward     NOECHO     #WM SCROLL FORWARD CMS 1
   CMSPF 09 Rdrlist     ECHO       EXEC RDRLIST
   CMSPF 10 Left        NOECHO     #WM SCROLL LEFT CMS 10
   CMSPF 11 Right       NOECHO     #WM SCROLL RIGHT CMS 10
   CMSPF 12 Cmdline     NOECHO     CURSOR VSCREEN CMS -2 8 (RES
   CMSPF 13 Help        ECHO       HELP
   CMSPF 14 Pop_Msg     NOECHO     #WM POP WINDOW MESSAGE *
   CMSPF 15 Quit        NOECHO     SET FULLSCREEN SUSPEND

   PF1=Help      2=Pop_Msg   3=Quit       4=Clear_Top  5=Filelist   6=Retrieve
   PF7=Backward  8=Forward   9=Rdrlist   10=Left      11=Right     12=Cmdline
   ====> _
  15:32:11                             Enter a command or press a PF or PA key
```

Figure 38. Displaying the CMSPF Key Settings

To scroll through your CMSPF key settings in full-screen CMS, you can use your CMSPF keys. CMSPF 7 scrolls backward one window display. CMSPF 8 scrolls forward one window display.

The #WM Command

When you displayed the CMSPF key settings in the example above, you may have noticed that several of the CMSPF key settings contain the Immediate command "#WM." By using #WM in the PF key definition, you can set CMSPF keys to perform windowing commands which will be executed immediately in the CMS window. You can set your CMSPF keys without using #WM, however, your PF key commands may be executed after other commands which are pending at the time you press the PF key. You can use #WM to set PF keys to perform any of the windowing commands listed below:

CLEAR WINDOW	PUT SCREEN	RESTORE WINDOW
CP	QUERY BORDER	SCROLL
DROP WINDOW	QUERY HIDE	SET BORDER
HIDE WINDOW	QUERY LOCATION	SET LOCATION
MAXIMIZE WINDOW	QUERY RESERVED	SET RESERVED
MINIMIZE WINDOW	QUERY SHOW	SET WINDOW
POP WINDOW	QUERY WINDOW	SET WMPF
POSITION WINDOW	QUERY WMPF	

You can also issue #WM commands from the command line or anywhere else in the CMS window.

Setting a CMSPF Key

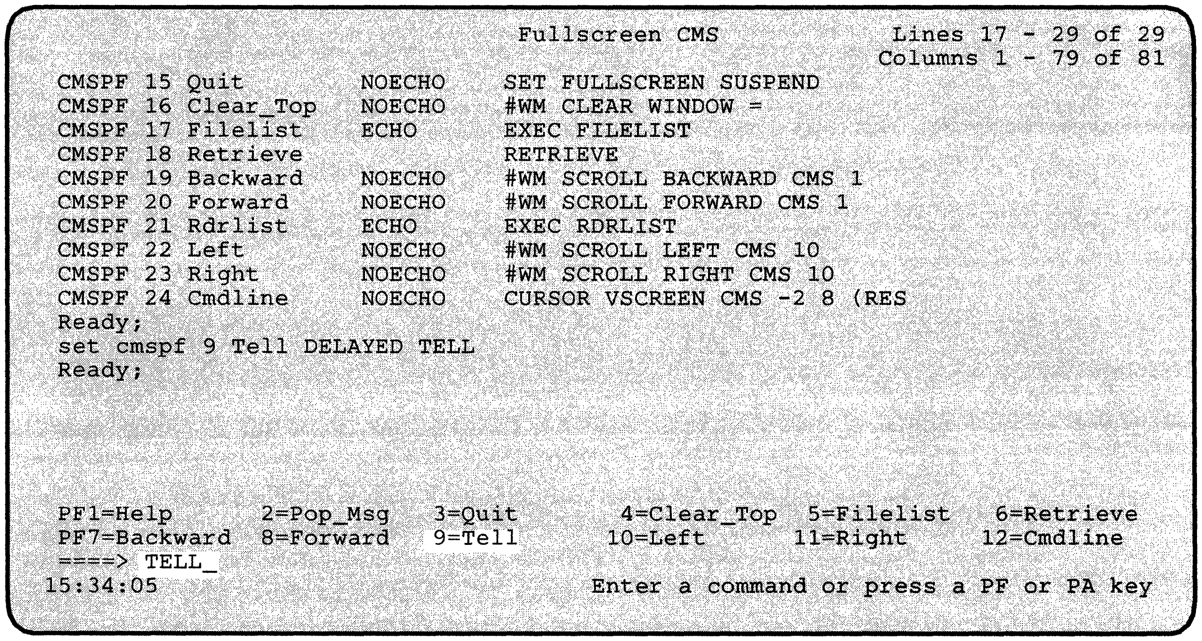
You can change the setting of a CMSPF key by issuing SET CMSPF followed by the PF key number (represented as nn) and the pseudonym, optional keyword, and associated command.

Let's reset a PF key, PF9, to the TELL command. Then, whenever you want to send a message, you can press PF9, and then type a userid or nickname and message. Type the command:

```
set cmspf 9 Tell DELAYED TELL
```

and press the ENTER key.

Your CMSPF 9 key pseudonym at the bottom of the physical screen should show that it is assigned the pseudonym "Tell." Now press PF9. TELL appears on the command line like this:



```
                                 Fullscreen CMS          Lines 17 - 29 of 29
                                                        Columns 1 - 79 of 81
 CMSPF 15 Quit        NOECHO    SET FULLSCREEN SUSPEND
 CMSPF 16 Clear_Top   NOECHO    #WM CLEAR WINDOW =
 CMSPF 17 Filelist    ECHO      EXEC FILELIST
 CMSPF 18 Retrieve              RETRIEVE
 CMSPF 19 Backward    NOECHO    #WM SCROLL BACKWARD CMS 1
 CMSPF 20 Forward     NOECHO    #WM SCROLL FORWARD CMS 1
 CMSPF 21 Rdrlist     ECHO      EXEC RDRLIST
 CMSPF 22 Left        NOECHO    #WM SCROLL LEFT CMS 10
 CMSPF 23 Right       NOECHO    #WM SCROLL RIGHT CMS 10
 CMSPF 24 Cmdline     NOECHO    CURSOR VSCREEN CMS -2 8 (RES
 Ready;
 set cmspf 9 Tell DELAYED TELL
 Ready;




 PF1=Help      2=Pop_Msg   3=Quit       4=Clear_Top  5=Filelist   6=Retrieve
 PF7=Backward  8=Forward   9=Tell      10=Left      11=Right     12=Cmdline
 ====> TELL_
15:34:05                                  Enter a command or press a PF or PA key
```

Figure 39. Setting CMSPF 9 to TELL

For complete descriptions of the CMSPF keys, refer to the SET CMSPF command in the *VM/SP CMS Command Reference.*

The PA1 Key

In addition to PF keys, you have a PA1 key on your keyboard which is assigned to POP WINDOW WM. The key may have different labels, depending on your terminal. If you do not have a key labelled PA1, ask your system administrator to show you the equivalent key.

PA1 pops the WM window. As discussed in the section entitled "The WM Window" on page 180, the WM window provides you with a command line and a set of PF keys for use in manipulating other windows.

The PA2 Key

The PA2 key works in the CMS window to scroll the top window displayed on your screen forward. PA2 is very useful for controlling windows which are automatically displayed on your screen such as the MESSAGE or WARNING window. When a MESSAGE or WARNING window appears on your screen, simply press PA2 to scroll the data forward. When there is no more data, the window will disappear from your screen.

In the WM window, the PA2 key will scroll the data in the topmost window. Once you have scrolled all the data, pressing PA2 will cause you to exit from the WM environment.

Depending on your terminal type, you may not have a key labelled PA2. Again, your system administrator should be able to show you the equivalent key.

The CLEAR Key

The CLEAR key performs the same function as the PA2 key. In the CMS window, the CLEAR key scrolls forward the topmost window displayed on your screen. In the WM window, the CLEAR key scrolls the topmost window and exits the WM environment when you have scrolled all the data. In both cases, pressing the CLEAR key causes the entire screen to be rewritten.

Once again, depending on your terminal, you may not have a key labelled CLEAR. Your system administrator should be able to show you the equivalent key.

The WMPF Keys

The PF keys in the WM environment are different from the CMSPF key settings. The WM settings are:

WMPF Key	Pseudonym	Command
WMPF 1	Help	NOECHO HELP
WMPF 2	Top	NOECHO SCROLL TOP =
WMPF 3	Quit	NOECHO DROP WINDOW WM
WMPF 4	Clear	NOECHO CLEAR WINDOW =
WMPF 5	Copy	NOECHO PUT SCREEN COPY SCREEN
WMPF 6	Retrieve	RETRIEVE
WMPF 7	Backward	NOECHO SCROLL BACKWARD = 1
WMPF 8	Forward	NOECHO SCROLL FORWARD = 1
WMPF 9	Maximize	NOECHO MAXIMIZE WINDOW =
WMPF 10	Left	NOECHO SCROLL LEFT = 10
WMPF 11	Right	NOECHO SCROLL RIGHT = 10
WMPF 12	Restore	NOECHO RESTORE WINDOW =

Figure 40. WMPF Key Settings

Note that for terminals equipped with 24 PF keys, PF keys 13 through 24 have the same values as PF keys 1 through 12, respectively.

Messages in Full-Screen CMS

If you are already familiar with sending and receiving messages on the system, you will find the MESSAGE window helpful in full-screen CMS. Through this window, you can view messages without clearing the physical screen and your work will not be interrupted. (If you are not in full-screen CMS, the screen is cleared when you press ENTER to see a message).

We'll assume you have created entries in your names file. If not, refer to Chapter 8, "Communicating with Other Computer Users," and create an entry for someone you work with. Then use a nickname for someone in your names file for the following exercise.

First, let's send a message. If you completed the previous exercise, you have "TELL" at the command line. Complete a message to your friend:

```
TELL debbie Send me a message.
```

If your friend does not respond with a message, try again with someone else in your names file. When you receive a message, you will be notified as follows:

- the terminal alarm will sound
- the status area message class indicator will be updated to show that you have received a message.

Now press the ENTER key to see the message in the MESSAGE window. The MESSAGE window will pop.

Note: The MESSAGE window must contain at least one message before the window will be displayed.

Dropping and Popping a Window

There are many ways to manage your MESSAGE window. The easiest way is to use the PA2 key. When the MESSAGE window pops, you can press PA2 to scroll the window. When you have seen all the messages in the window, pressing PA2 again will cause the window to disappear from your screen.

You can also use the POP WINDOW and DROP WINDOW commands to view the MESSAGE window or remove it from the physical screen.

The MESSAGE window is variable in size, that is, it expands as more messages are received. If you have not received any messages, the window will not be displayed. Once you receive a message, the window will be displayed and will expand as you receive more messages.

Let's try removing the previous message from your screen by entering:

```
drop window message
```

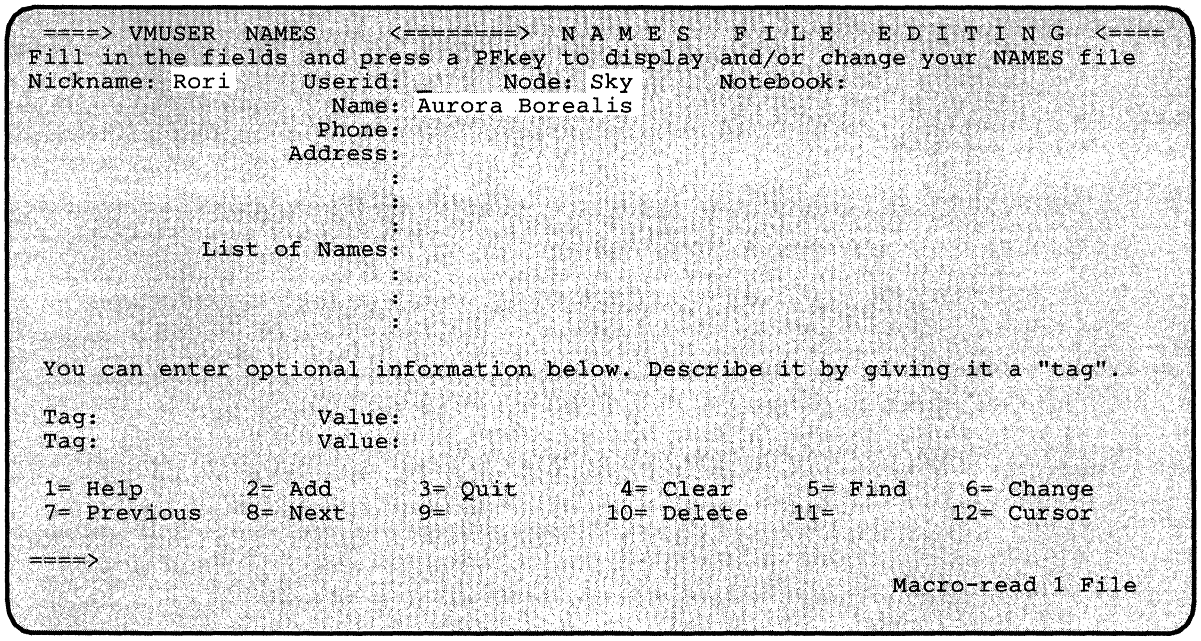
If you want to redisplay the MESSAGE window, press CMSPF 2 which is assigned to pop the MESSAGE window or enter POP WINDOW MESSAGE.

Working with the Names File

The MESSAGE window is very useful when you are editing a file and need to ask someone for information. To show you how this works, follow this example to add a new entry to your names file. First enter the command:

```
names
```

When the names panel appears on your screen, begin filling in the following information as shown:



```
 ====> VMUSER  NAMES     <=========>  N A M E S   F I L E   E D I T I N G  <====
Fill in the fields and press a PFkey to display and/or change your NAMES file
Nickname: Rori      Userid: _      Node: Sky      Notebook:
                      Name: Aurora Borealis
                     Phone:
                   Address:
                          :
                          :
                          :
             List of Names:
                          :
                          :
                          :

 You can enter optional information below. Describe it by giving it a "tag".

 Tag:                Value:
 Tag:                Value:

 1= Help       2= Add       3= Quit       4= Clear     5= Find     6= Change
 7= Previous   8= Next      9=           10= Delete   11=         12= Cursor

====>
                                                              Macro-read 1 File
```

Figure 41. Adding an Entry to the Names File

Suppose that you suddenly realized you didn't know the userid. Move the cursor to the command line, and send a message to a friend:

```
tell babs What's Rori's userid?
```

When your friend sends a reply to your message, the terminal alarm sounds. Press ENTER to display the MESSAGE window. The window appears on top of the names file like this:

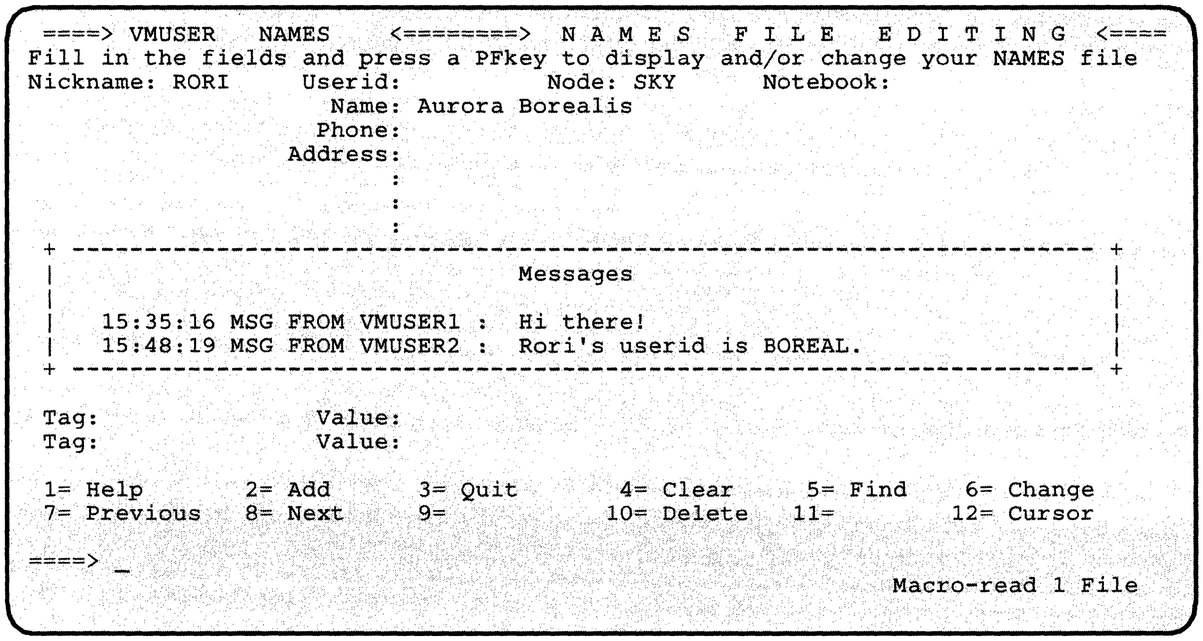

```
 ====> VMUSER   NAMES    <========> N A M E S   F I L E   E D I T I N G  <====
Fill in the fields and press a PFkey to display and/or change your NAMES file
Nickname: RORI     Userid:          Node: SKY      Notebook:
                     Name: Aurora Borealis
                    Phone:
                  Address:
                         :
                         :
                         :
 + ------------------------------------------------------------------------ +
 |                              Messages                                    |
 |                                                                          |
 |   15:35:16 MSG FROM VMUSER1 :  Hi there!                                 |
 |   15:48:19 MSG FROM VMUSER2 :  Rori's userid is BOREAL.                  |
 + ------------------------------------------------------------------------ +

 Tag:                Value:
 Tag:                Value:

 1= Help       2= Add       3= Quit        4= Clear     5= Find     6= Change
 7= Previous   8= Next      9=            10= Delete   11=         12= Cursor

====> _
                                                            Macro-read 1 File
```

Figure 42. The MESSAGE Window in the Names File

You may notice that other messages you received since you set full-screen on are shown in the MESSAGE window. This will occur if you have not cleared the window by scrolling it forward or by issuing the command CLEAR WINDOW MESSAGE. If you have received several messages, you may need to issue the command SCROLL FORWARD in order to view your most recent message.

Now you can fill in the userid. When you are finished, drop the MESSAGE window by entering the DROP WINDOW MESSAGE command. You can also use the CLEAR WINDOW MESSAGE command which will scroll the window forward past the current messages and remove the window from your screen. The CLEAR WINDOW MESSAGE command will also position the window so that when a new message is received, the message will appear at the top of the window.

Once you have dropped the MESSAGE window, you can then add the entry to your names directory (with PF2) and exit the names file (with PF3).

Re-Entering Commands

Full-screen CMS provides you with several ways to easily re-enter commands you issued previously. You can press the CMSPF 6 key, which is set to RETRIEVE. You can also scroll back through your CMS session and re-type and re-issue commands you entered previously.

Using the RETRIEVE Key

When you first press CMSPF 6, the latest command you entered will be redisplayed on the command line. If you press it again, the previous line will be displayed. If you continue to press CMSPF 6, the commands you entered previously will be displayed, one at a time.

When the command you wish to re-enter is displayed, you can simply press ENTER to execute the command again.

Entering Commands from the Screen

You have probably noticed that while working in full-screen CMS, each time you enter a command on the command line, the command you typed remains on the physical screen. For example, if you completed the previous exercises, you may have the following commands on your screen:

```
Ready;
set cmspf 9 Tell DELAYED TELL
Ready;
tell debbie Send me a message.
Ready;
drop window message
Ready;
names
Ready;
```

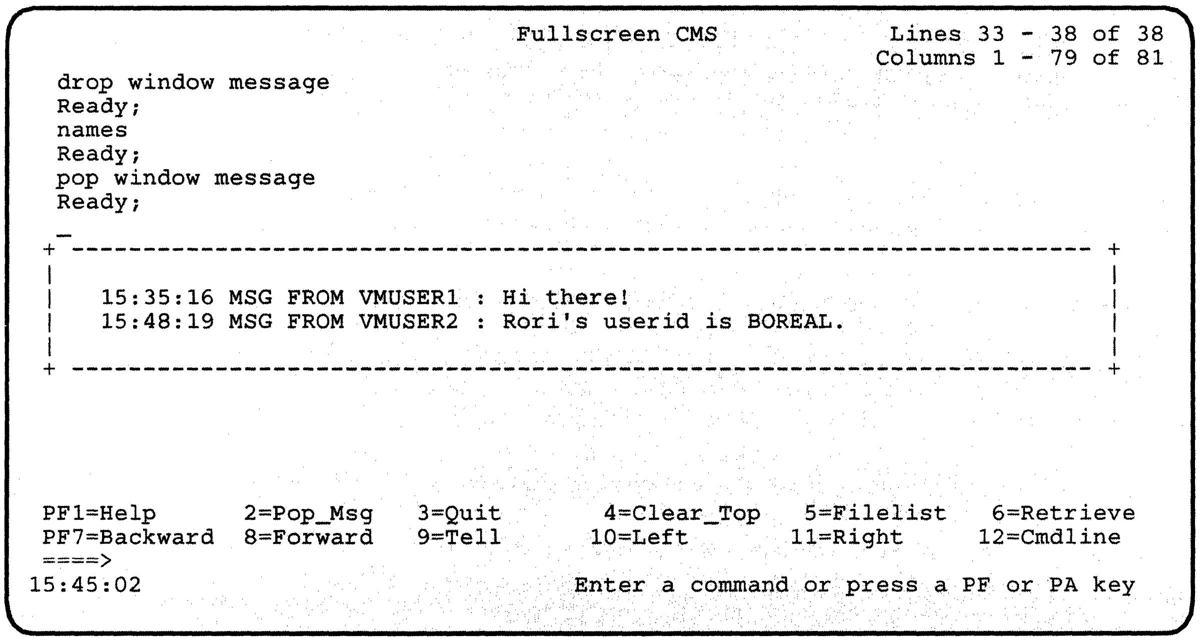
With full-screen CMS, you can re-enter any of these commands by moving your cursor to the place on the screen where the command is written, typing over at least one character, then pressing the ENTER key. You can also change a word or words of a command that you previously entered, then re-enter the new command.

For example, move your cursor under the command DROP WINDOW MESSAGE which currently appears on your screen. Type "POP" over "DROP." Press ENTER to issue the command:

```
pop window message
```

from the same line on your physical screen.

The MESSAGE window will reappear on your screen as follows:



```
                                 Fullscreen CMS           Lines 33 - 38 of 38
                                                          Columns 1 - 79 of 81
 drop window message
 Ready;
 names
 Ready;
 pop window message
 Ready;
 _
+ --------------------------------------------------------------------------- +
|                                                                             |
|   15:35:16 MSG FROM VMUSER1 : Hi there!                                     |
|   15:48:19 MSG FROM VMUSER2 : Rori's userid is BOREAL.                      |
|                                                                             |
+ --------------------------------------------------------------------------- +




PF1=Help      2=Pop_Msg   3=Quit       4=Clear_Top   5=Filelist    6=Retrieve
PF7=Backward  8=Forward   9=Tell      10=Left       11=Right      12=Cmdline
====>
15:45:02                              Enter a command or press a PF or PA key
```

Figure 43. Popping the MESSAGE Window

To drop the MESSAGE, move your cursor under the command DROP WINDOW MESSAGE. Re-type any letter and press ENTER.

Logging Messages and Other Information

When you issue the command SET FULLSCREEN ON, by default, messages and warnings will be logged for you. Messages will be logged into a file with the filename and filetype of MESSAGE LOGFILE; warnings will be logged into WARNING LOGFILE.

To view all the messages you sent or received during your terminal session, you would simply need to XEDIT or print the file MESSAGE LOGFILE. To view warnings you have received, you would XEDIT or print the file WARNING LOGFILE.

If you wish, you can use the SET LOGFILE command to log your CMS output and other information into separate files which you can later XEDIT or print. For details on how to log information, refer to the SET LOGFILE command in the *VM/SP CMS Command Reference.*

Working with Border Commands

You have already learned enough about full-screen CMS to work with windows. But there's another feature that could make working with windows even easier. Single character commands can be typed in the corner of a window border. These commands are called Border commands. You can scroll, move, drop or clear a window by entering a letter in the border corner.

Borders are optional and can be set on or off. (See the DEFINE WINDOW and SET BORDER commands in the *VM/SP CMS Command Reference* for more information). Because borders frame a window, if the window is the same size as the physical screen, or if it's positioned in such a way that the borders do not fit on the physical screen, the borders will not be shown. For the following examples, we will use windows with predefined borders that fit within the physical screen.

In trying some Border commands, let's look at the MESSAGE window again. Because the MESSAGE window is variable in size, you will notice that the window size changes as we go through the examples.

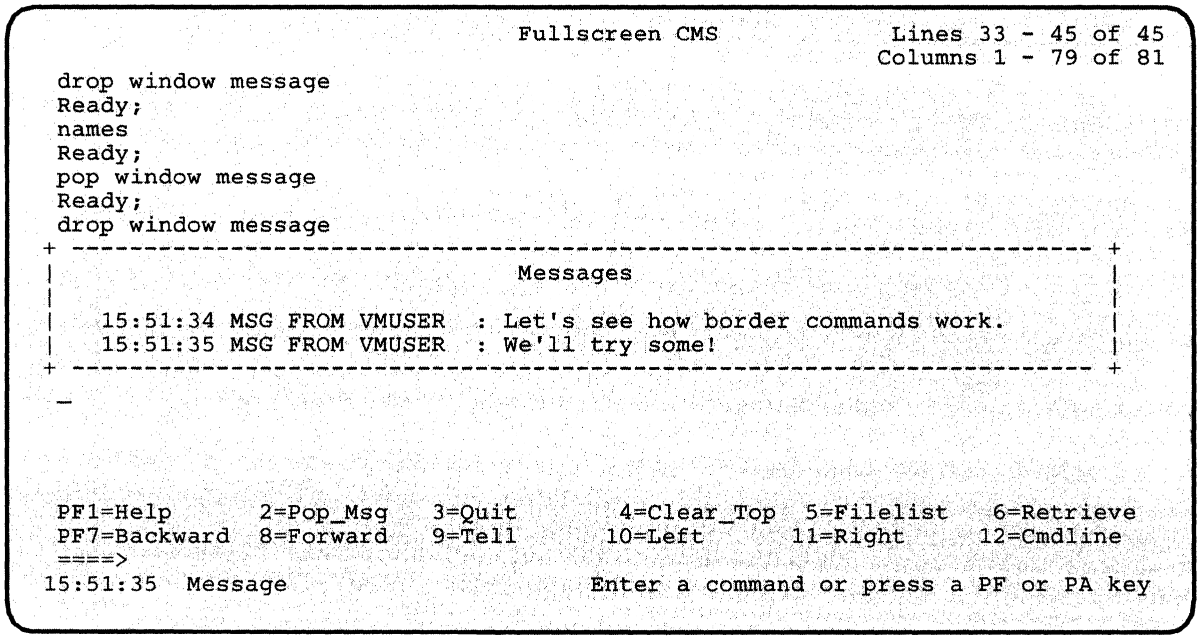
First, clear the message virtual screen of all old messages by entering:

```
clear vscreen message
```

Next, send the following two messages to yourself. Type a '#' (the default line end character) between sentences.

```
tell * Let's see how border commands work.#tell * We'll try some!
```

Finally, press the ENTER key (once). Your messages now appear in the MESSAGE window. The corners of the window border are represented by a plus (+) sign.

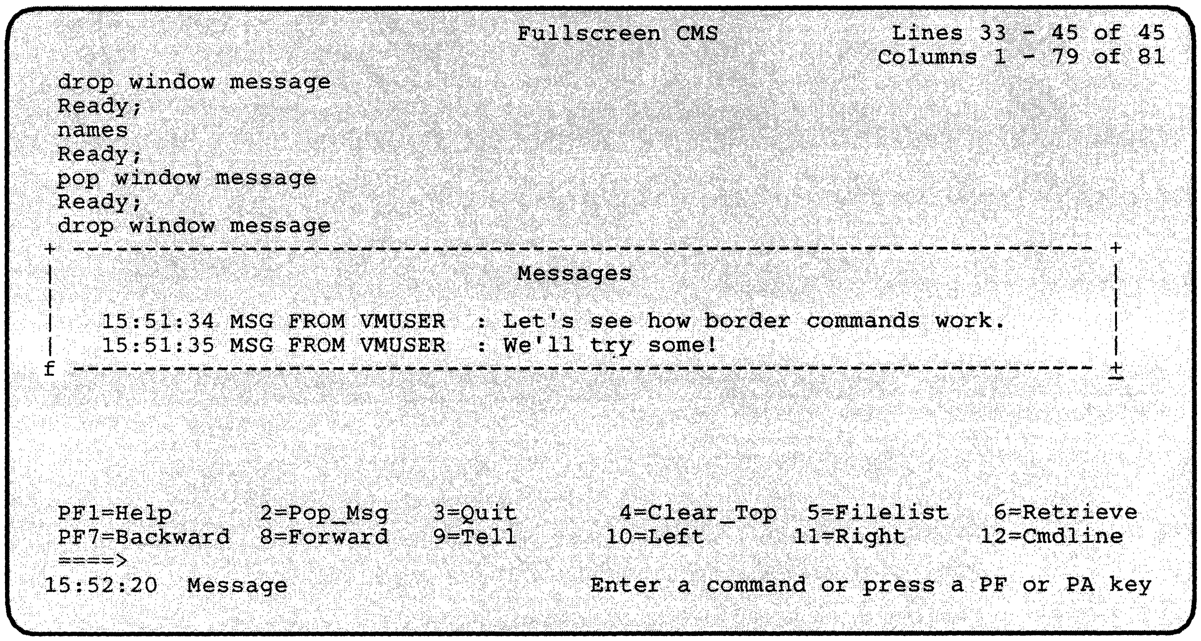


```
                              Fullscreen CMS            Lines 33 - 45 of 45
                                                       Columns 1 - 79 of 81
 drop window message
 Ready;
 names
 Ready;
 pop window message
 Ready;
 drop window message
+ ------------------------------------------------------------------------ +
|                               Messages                                   |
|                                                                          |
|   15:51:34 MSG FROM VMUSER  : Let's see how border commands work.        |
|   15:51:35 MSG FROM VMUSER  : We'll try some!                            |
+ ------------------------------------------------------------------------ +
 _



 PF1=Help      2=Pop_Msg   3=Quit       4=Clear_Top  5=Filelist   6=Retrieve
 PF7=Backward  8=Forward   9=Tell      10=Left      11=Right     12=Cmdline
 ====>
15:51:35  Message                     Enter a command or press a PF or PA key
```

Figure 44. Looking at the Corners of a Window Border

Scrolling Forward and Backward

Now, to scroll the window forward, type the letter "F" in any corner like this:

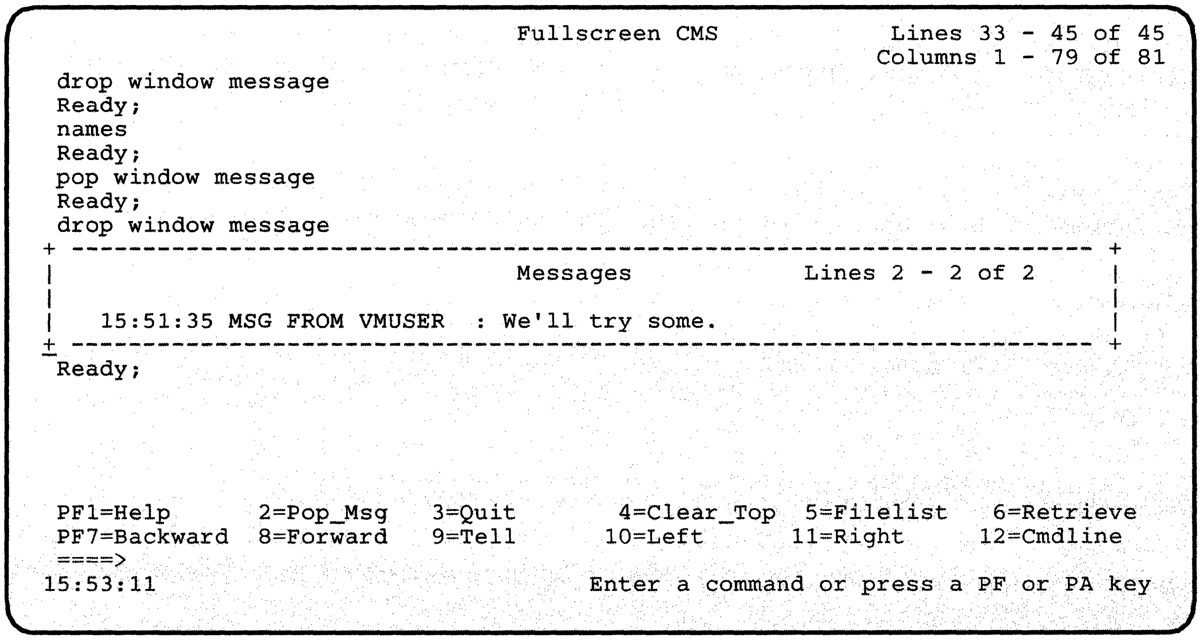


```
                                    Fullscreen CMS            Lines 33 - 45 of 45
                                                              Columns 1 - 79 of 81
 drop window message
 Ready;
 names
 Ready;
 pop window message
 Ready;
 drop window message
+ ------------------------------------------------------------------------ +
|                                   Messages                                |
|                                                                           |
|   15:51:34 MSG FROM VMUSER  : Let's see how border commands work.         |
|   15:51:35 MSG FROM VMUSER  : We'll try some!                             |
f ------------------------------------------------------------------------ +




 PF1=Help      2=Pop_Msg   3=Quit       4=Clear_Top  5=Filelist   6=Retrieve
 PF7=Backward  8=Forward   9=Tell      10=Left      11=Right     12=Cmdline
 ====>
15:52:20  Message                      Enter a command or press a PF or PA key
```

Figure 45. Using a Border Command to Scroll Forward

and press the ENTER key. Notice that the last line displayed becomes the first and only line displayed. The window size also changes because the MESSAGE window is variable in size. The window grows or shrinks depending on how much data there is to display.

The following example shows the result of scrolling forward.

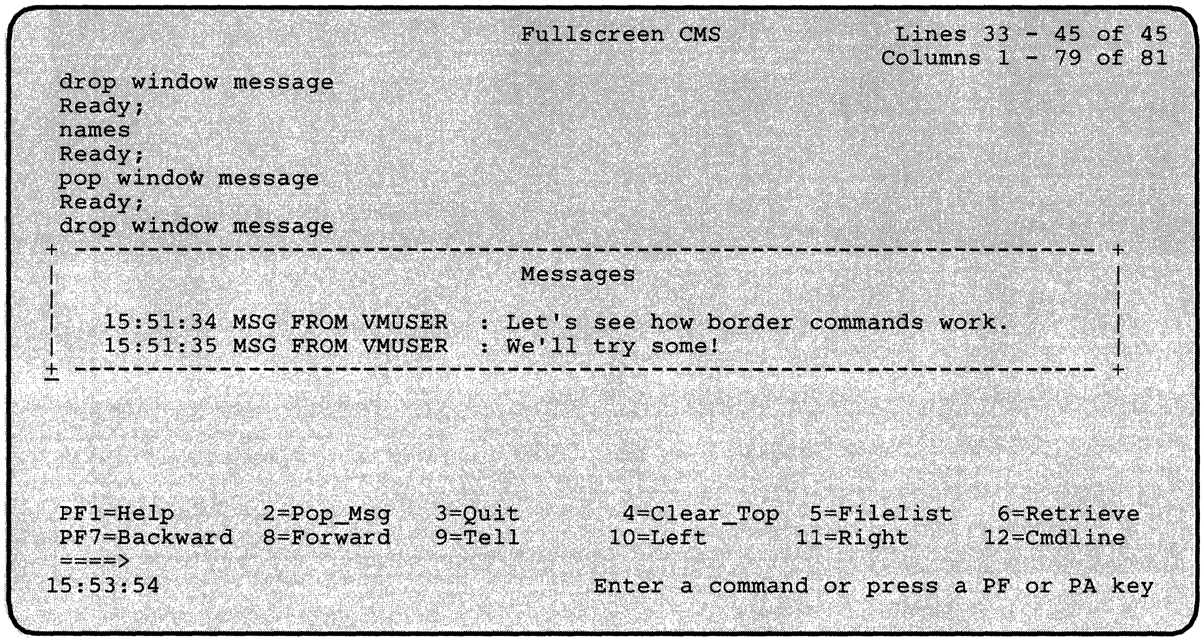


```
                               Fullscreen CMS           Lines 33 - 45 of 45
                                                      Columns 1 - 79 of 81
 drop window message
 Ready;
 names
 Ready;
 pop window message
 Ready;
 drop window message
+ ------------------------------------------------------------------------- +
|                              Messages            Lines 2 - 2 of 2         |
|                                                                           |
|   15:51:35 MSG FROM VMUSER  : We'll try some.                             |
+ ------------------------------------------------------------------------- +
 Ready;





 PF1=Help      2=Pop_Msg   3=Quit       4=Clear_Top  5=Filelist   6=Retrieve
 PF7=Backward  8=Forward   9=Tell      10=Left      11=Right     12=Cmdline
 ====>
15:53:11                                 Enter a command or press a PF or PA key
```

Figure 46. The Result of Scrolling Forward

Now that you have viewed all the data in the MESSAGE window, if you scrolled forward again, the window would disappear from your screen.

Let's scroll the same window backward. Enter a "B"in any corner, and press ENTER. Assuming you did not receive any new messages, the window now looks like this:



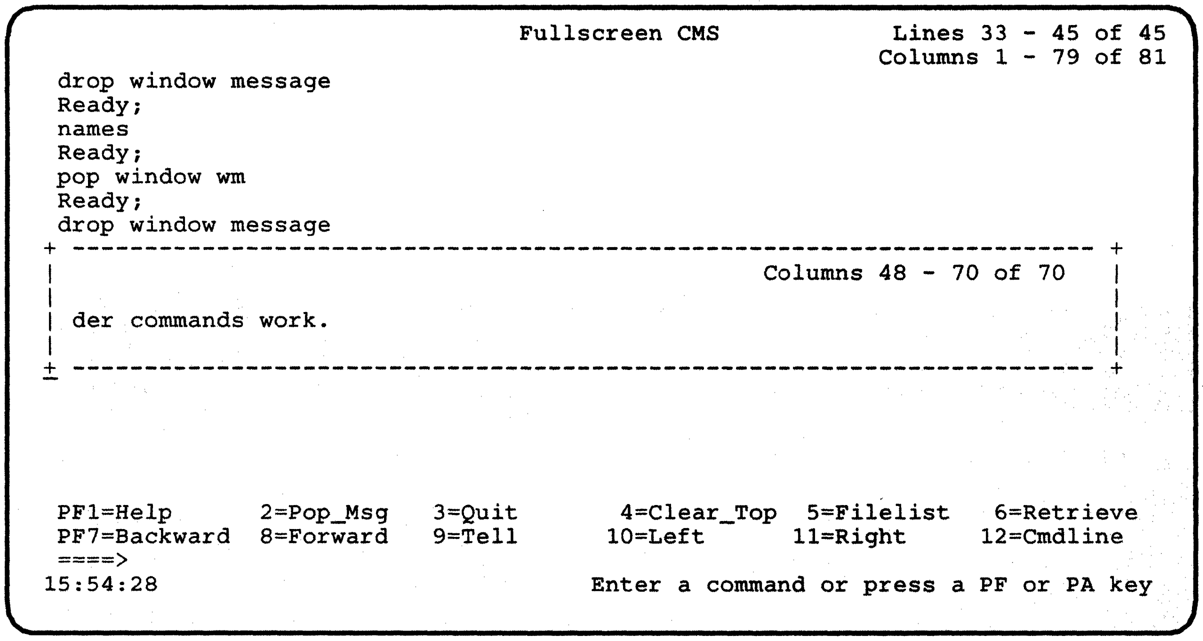
```
                              Fullscreen CMS            Lines 33 - 45 of 45
                                                        Columns 1 - 79 of 81
 drop window message
 Ready;
 names
 Ready;
 pop window message
 Ready;
 drop window message
+ ---------------------------------------------------------------------- +
|                             Messages                                   |
|                                                                        |
|   15:51:34 MSG FROM VMUSER  : Let's see how border commands work.      |
|   15:51:35 MSG FROM VMUSER  : We'll try some!                          |
+ ---------------------------------------------------------------------- +




 PF1=Help      2=Pop_Msg   3=Quit      4=Clear_Top  5=Filelist   6=Retrieve
 PF7=Backward  8=Forward   9=Tell     10=Left      11=Right     12=Cmdline
 ====>
15:53:54                              Enter a command or press a PF or PA key
```

Figure 47. Scrolling Backward through a Window Border

Scrolling Right and Left

Next, let's try scrolling the window to the right and left. Enter an "R" in a corner, and press ENTER to move the window to the right. Notice the location information, "Columns 48 - 70 of 70" which appears within the window. This indicates that the data you are viewing in the window represents the right-most portion of the data available for viewing.

The window looks like this:



```
                                Fullscreen CMS          Lines 33 - 45 of 45
                                                        Columns 1 - 79 of 81
 drop window message
 Ready;
 names
 Ready;
 pop window wm
 Ready;
 drop window message
+ --------------------------------------------------------------------- +
|                                                Columns 48 - 70 of 70   |
|                                                                        |
| der commands work.                                                     |
|                                                                        |
+ --------------------------------------------------------------------- +




 PF1=Help      2=Pop_Msg   3=Quit       4=Clear_Top  5=Filelist   6=Retrieve
 PF7=Backward  8=Forward   9=Tell      10=Left      11=Right     12=Cmdline
 ====>
15:54:28                                  Enter a command or press a PF or PA key
```

Figure 48. Scrolling to the Right through a Window Border

Now, let's return the window to its previous position. Enter an "L" in a corner. The resulting window now looks like this:

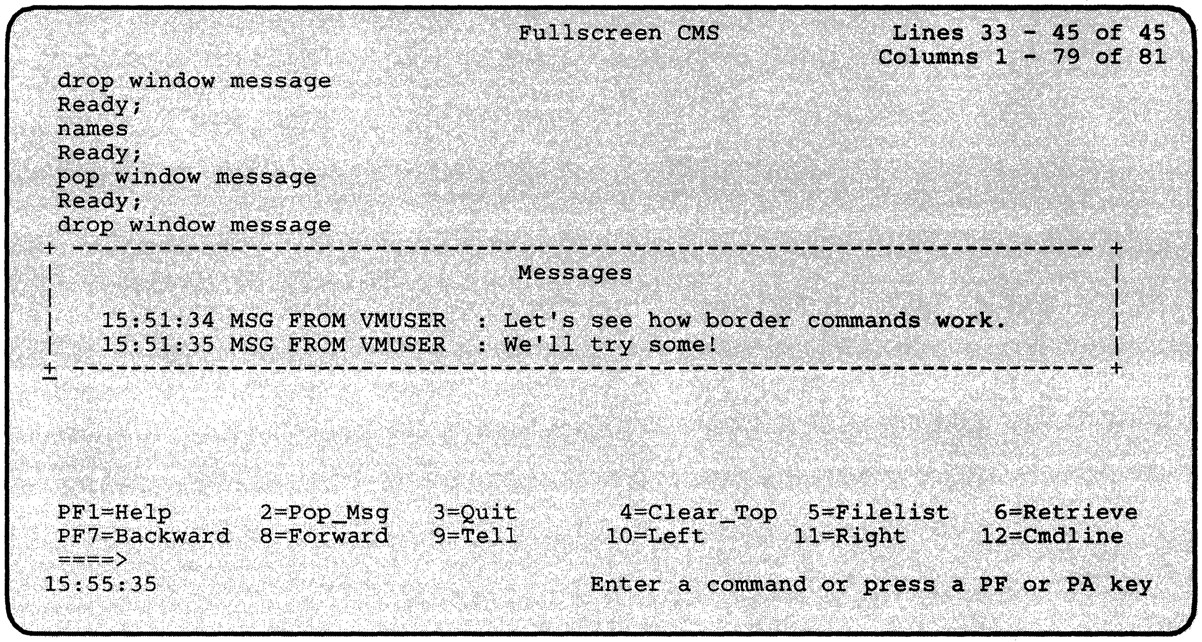

```
                                   Fullscreen CMS          Lines 33 - 45 of 45
                                                         Columns 1 - 79 of 81
  drop window message
  Ready;
  names
  Ready;
  pop window message
  Ready;
  drop window message
+ ------------------------------------------------------------------------ +
|                                  Messages                                 |
|                                                                           |
|   15:51:34 MSG FROM VMUSER  : Let's see how border commands work.        |
|   15:51:35 MSG FROM VMUSER  : We'll try some!                            |
+ ------------------------------------------------------------------------ +




  PF1=Help      2=Pop_Msg    3=Quit        4=Clear_Top  5=Filelist   6=Retrieve
  PF7=Backward  8=Forward    9=Tell       10=Left      11=Right     12=Cmdline
  ====>
 15:55:35                              Enter a command or press a PF or PA key
```

Figure 49. Scrolling to the Left through a Window Border

For further information on using Border commands, refer to the *VM/SP CMS Command Reference.*

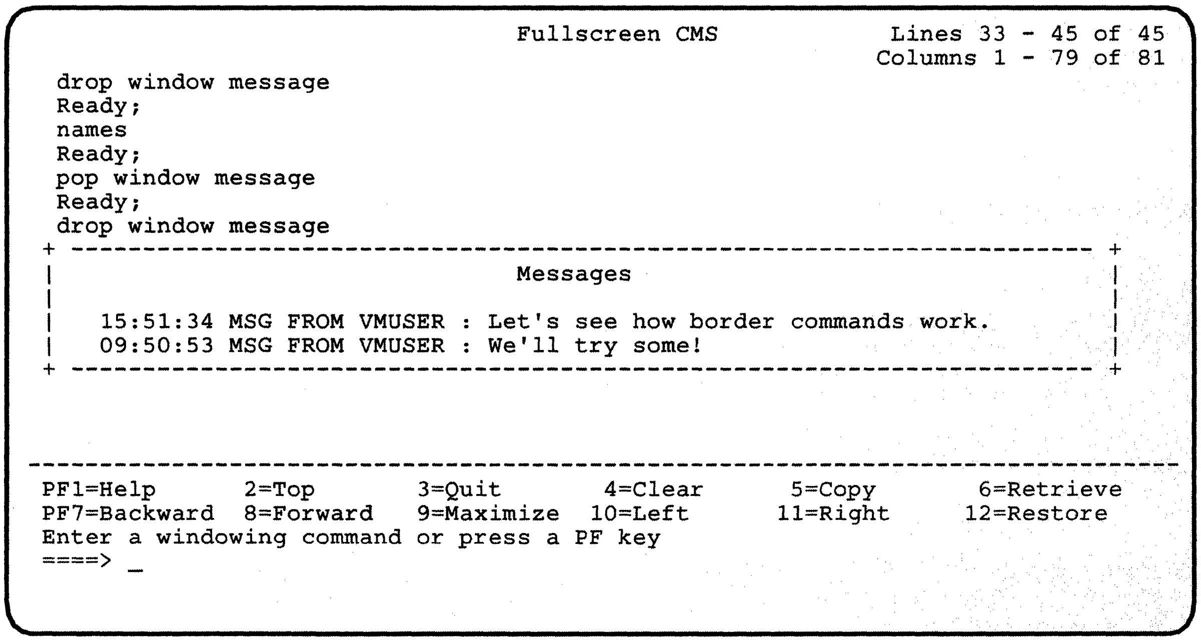
Using the WM Window

If you did not wish to use border commands to manipulate windows, you could, instead, press PA1 to pop the WM window and use the WMPF keys to perform these same functions. The WM window is useful for manipulating the topmost window showing on your screen.

If you have followed the exercises above, the MESSAGE window is currently showing on your screen. Press PA1 to pop the WM window.

Note: If you receive a message regarding the SET FULLREAD command, disregard the message.

Your screen will appear as shown below:



```
                              Fullscreen CMS          Lines 33 - 45 of 45
                                                    Columns 1 - 79 of 81
 drop window message
 Ready;
 names
 Ready;
 pop window message
 Ready;
 drop window message
+ ---------------------------------------------------------------------- +
|                             Messages                                   |
|                                                                        |
|   15:51:34 MSG FROM VMUSER : Let's see how border commands work.      |
|   09:50:53 MSG FROM VMUSER : We'll try some!                          |
+ ---------------------------------------------------------------------- +



-------------------------------------------------------------------------------
PF1=Help      2=Top       3=Quit      4=Clear      5=Copy       6=Retrieve
PF7=Backward  8=Forward   9=Maximize  10=Left      11=Right     12=Restore
Enter a windowing command or press a PF key
====> _
```

Figure 50. Popping the WM Window

If you wanted to scroll the MESSAGE window forward, one way to do so would be to use the "F" border command, as we did in the previous exercises. However, you could also use WMPF 8 which will scroll the window forward.

Press WMPF 8 to scroll the MESSAGE window forward. Press it again to scroll to the bottom of the window and remove the window from your screen.

Your screen will look like this:

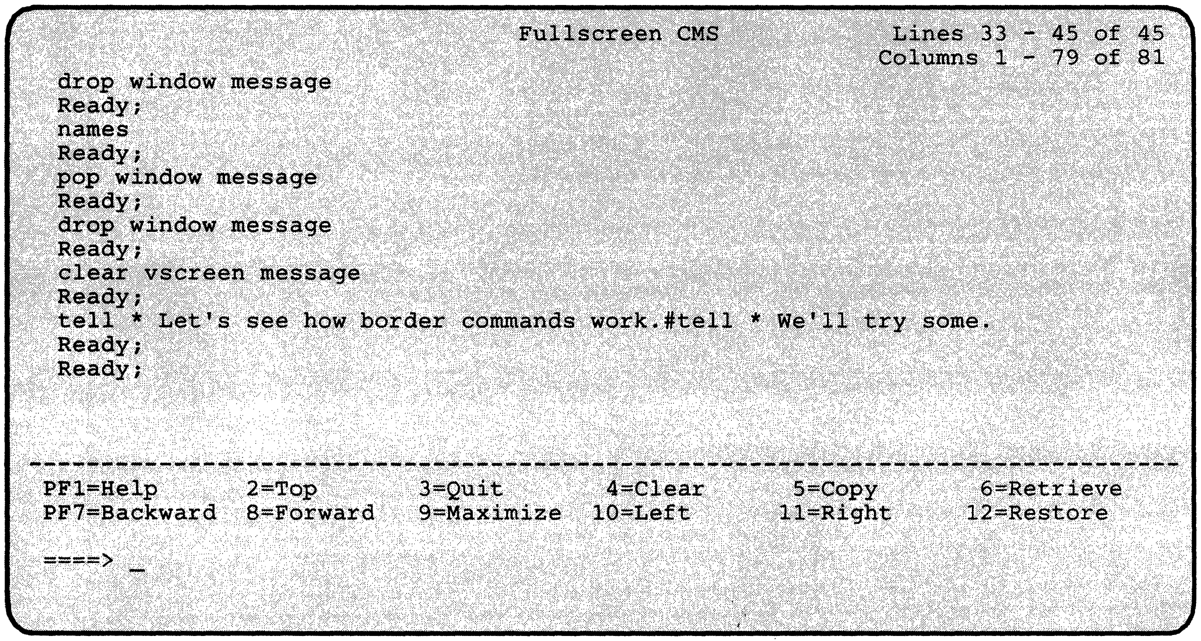

```
                               Fullscreen CMS           Lines 33 - 45 of 45
                                                       Columns 1 - 79 of 81
  drop window message
  Ready;
  names
  Ready;
  pop window message
  Ready;
  drop window message
  Ready;
  clear vscreen message
  Ready;
  tell * Let's see how border commands work.#tell * We'll try some.
  Ready;
  Ready;



--------------------------------------------------------------------------------
 PF1=Help      2=Top       3=Quit      4=Clear      5=Copy       6=Retrieve
 PF7=Backward  8=Forward   9=Maximize  10=Left      11=Right     12=Restore

 ====> _
```

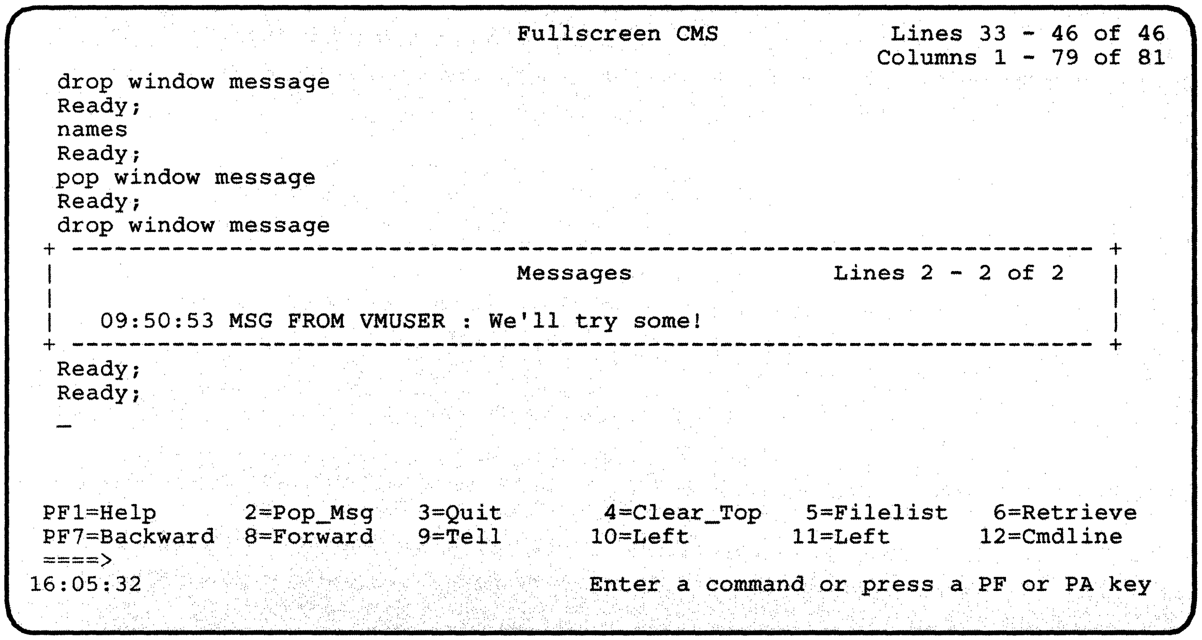
Figure 51. Dropping the MESSAGE Window

Now, only the WM window remains on your screen. Press WMPF 3 to drop the WM window.

You could also use other WMPF keys to manipulate windows. For example, WMPF 10 performs the same function as the "L" border command we used previously; WMPF 11 performs the same function as "R."

Now, to show you another unique feature of the WM window, we will purposely create a situation where the window will pop automatically. First, press CMSPF 2 to pop the MESSAGE window.

The window will appear as follows:



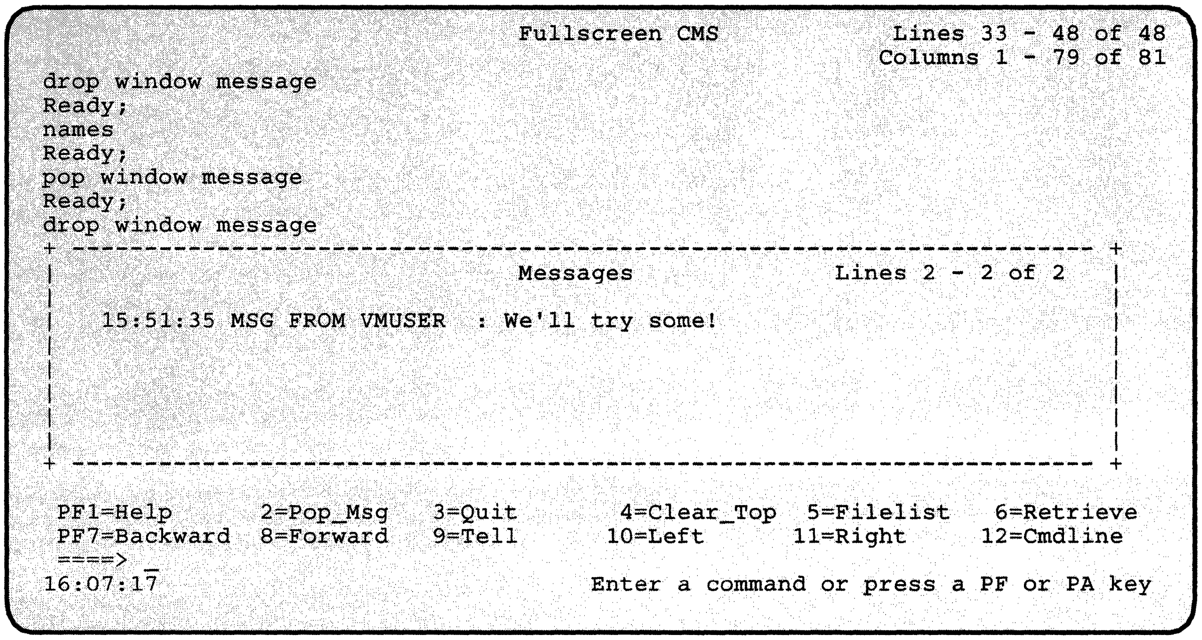
```
                               Fullscreen CMS          Lines 33 - 46 of 46
                                                     Columns 1 - 79 of 81
 drop window message
 Ready;
 names
 Ready;
 pop window message
 Ready;
 drop window message
+ ------------------------------------------------------------------- +
|                              Messages            Lines 2 - 2 of 2   |
|                                                                     |
|   09:50:53 MSG FROM VMUSER : We'll try some!                        |
+ ------------------------------------------------------------------- +
 Ready;
 Ready;

 _



PF1=Help      2=Pop_Msg   3=Quit      4=Clear_Top   5=Filelist   6=Retrieve
PF7=Backward  8=Forward   9=Tell     10=Left       11=Left      12=Cmdline
====>
16:05:32                             Enter a command or press a PF or PA key
```

Figure 52. Displaying the MESSAGE Window

You will see only the second message you received because in the previous exercise, WMPF 8 scrolled the window. Type the following command on the command line:

```
set window message fixed
```

Your screen will now look like this:

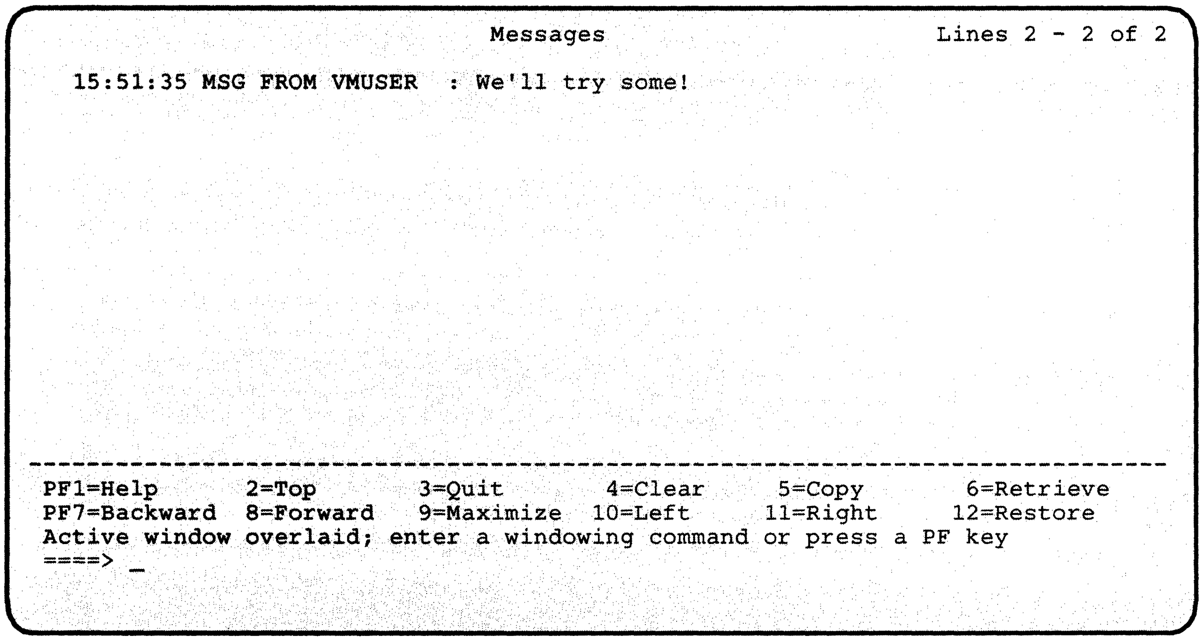


```
                          Fullscreen CMS              Lines 33 - 48 of 48
                                                      Columns 1 - 79 of 81
drop window message
Ready;
names
Ready;
pop window message
Ready;
drop window message
+ ---------------------------------------------------------------------- +
|                          Messages             Lines 2 - 2 of 2        |
|                                                                        |
|   15:51:35 MSG FROM VMUSER  : We'll try some!                          |
|                                                                        |
|                                                                        |
|                                                                        |
|                                                                        |
|                                                                        |
+ ---------------------------------------------------------------------- +

 PF1=Help      2=Pop_Msg   3=Quit       4=Clear_Top  5=Filelist   6=Retrieve
 PF7=Backward  8=Forward   9=Tell      10=Left      11=Right     12=Cmdline
 ====> _
16:07:17                              Enter a command or press a PF or PA key
```

Figure 53. Changing the MESSAGE Window

Now, enter the border command "X" from any corner of the MESSAGE window. The "X" command maximizes the window, that is, it enlarges the window to allow you to view more data.

Your screen looks like this:



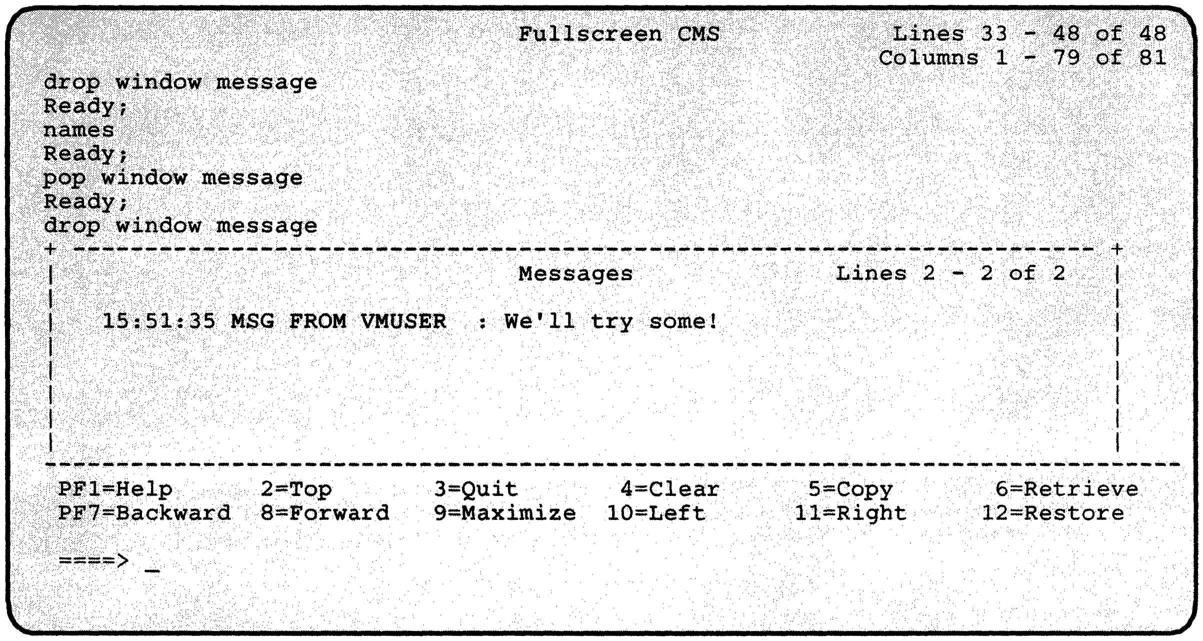
```
                        Messages                        Lines 2 - 2 of 2
  15:51:35 MSG FROM VMUSER  : We'll try some!









--------------------------------------------------------------------------------
PF1=Help      2=Top       3=Quit       4=Clear      5=Copy       6=Retrieve
PF7=Backward  8=Forward   9=Maximize  10=Left      11=Right     12=Restore
Active window overlaid; enter a windowing command or press a PF key
====> _
```

Figure 54. The WM Window

You will notice that the MESSAGE window is maximized, and the WM window popped automatically. The WM window pops because the maximized MESSAGE window is so large that it is covering up the screen, command line, and PF key settings. The WM window provides you with an area to enter commands to manipulate the window which is covering up your screen.

At this point, you could use the WMPF keys or enter windowing commands from the command line in the WM window to manipulate the maximized MESSAGE window. We will use one of the WMPF keys. Press WMPF 12 to restore the MESSAGE window.

The MESSAGE window appears as shown below:



```
                              Fullscreen CMS             Lines 33 - 48 of 48
                                                         Columns 1 - 79 of 81
drop window message
Ready;
names
Ready;
pop window message
Ready;
drop window message
+ ----------------------------------------------------------------------- +
|                             Messages                 Lines 2 - 2 of 2   |
|                                                                          |
|   15:51:35 MSG FROM VMUSER  : We'll try some!                           |
|                                                                          |
|                                                                          |
|                                                                          |
|                                                                          |
|                                                                          |
---------------------------------------------------------------------------------
 PF1=Help      2=Top        3=Quit      4=Clear      5=Copy      6=Retrieve
 PF7=Backward  8=Forward    9=Maximize  10=Left      11=Right    12=Restore

 ====> _
```

Figure 55. Restoring the MESSAGE Window

One way to exit from the WM environment is to press WMPF 3. This will remove both the WM and MESSAGE windows from your screen.

Even though you cannot see the MESSAGE window, let's reset it. Enter the following command:

```
set window message variable
```

to reset the MESSAGE window to its default status.

If you want to leave full-screen CMS, issue the command SET FULLSCREEN OFF. You can also suspend full-screen CMS by entering SET FULLSCREEN SUSPEND or pressing CMSPF 3.

Now that you are familiar with using full-screen CMS and windows to display information, you may wish to refer to Chapter 15, "Customizing Full-Screen CMS" for further information on ways to tailor full-screen CMS and windowing support for your particular needs.

You may also wish to refer to Appendix D, "Considerations for Full-Screen CMS and Windowing." This section contains information you may find useful when you begin using windowing and full-screen CMS.

Chapter 10. Using the CMS Batch Facility

The CMS batch facility provides a way of submitting jobs for batch processing in CMS. You can use the CMS batch facility when:

- You have a job (like an assembly or execution) that takes a lot of time, and you want to be able to use your terminal for other work while the time-consuming job is being run.

- You do not have access to a terminal.

The CMS batch facility is really a virtual machine, generated and controlled by the system operator, who logs on VM/SP using the batch userid and invoking the CMSBATCH facility by either:

- Entering the BATCH parameter in the PARM field of the IPL command, or

- Specifying the NOSPROF parameter and entering CMSBATCH at the VM READ.

All jobs submitted for batch processing are spooled to the userid of this virtual machine, which executes the jobs sequentially. To use the CMS batch facility at your location, you must ask the system operator what the userid of the batch virtual machine is.

Submitting Jobs to the CMS Batch Facility

Under a real OS or DOS system, jobs submitted in batch mode are controlled by JCL specifications. Batch jobs submitted to the CMS batch facility are controlled by the control cards /JOB, /SET, and /*, and by CMS commands.

Any application or development program written in a language supported by VM/SP may be executed on the batch facility virtual machine. However, there are restrictions on programs using certain CP and CMS commands, as described later in this section.

Input to the Batch Machine

Input records must be in card-image format, and may be punched on real cards, placed in a CMS file with fixed-length, 80-character records, or punched to your virtual punch. These jobs are sent to the batch virtual machine in one of two ways:

- By reading the real punched card input into the system card reader, or
- By spooling your virtual punch to the virtual reader of the batch virtual machine.

When you submit a real card deck to the batch machine, the first card in the deck must be a CP ID card. The ID card takes the form:

```
ID  userid
```

where:

ID must begin in card column one and be separated from userid (the batch facility virtual machine userid) by one or more blanks.

For example, if your installation's batch virtual machine has a userid of BATCH1, you punch the card:

```
ID BATCH1
```

and place it in front of your deck.

When you are going to submit a job using your virtual punch, you must first be sure that your punch is spooled to the virtual reader of the batch virtual machine:

```
spool punch to batch1
```

Submitting Virtual Card Input to the CMS Batch Facility

Virtual card input can be spooled to the batch machine in several ways. You may create a CMS file that contains the input control cards and use the CMS PUNCH command to punch the virtual cards:

```
punch batch jcl (noheader
```

When you punch a file this way, you must use the NOHEADER option of the PUNCH command, since the CMS batch facility cannot interpret the header card that is usually produced by the PUNCH command. As it does with cards in an invalid format, the batch virtual machine would flush the header card.

You can use an EXEC procedure to submit input to the batch machine. From an EXEC, you can punch one line at a time into your virtual punch, using the EXECIO command. When you do this, you must remember to use the CP CLOSE command to release the spool punch file when you are finished:

```
close punch
```

If you are using the EXEC to punch individual lines and entire CMS files to be read by the batch virtual machine as one continuous job stream, you must remember to spool your punch accordingly:

```
/* EXEC to submit a batch job to CMS BATCH */
spool punch cont
execio 1 punch '('string '/JOB MCGUIRE 999888'
punch batch jcl '* ('noheader
spool punch nocont
close punch
```

The /JOB and /* Cards

A /JOB card must precede each job to be executed under the batch facility. It identifies your userid to the batch virtual machine and provides accounting information for the system. It takes the form:

```
/JOB  userid accntnum [jobname] [comments]
```

where:

userid is your user identification, or the userid under which you want the job submitted. This parameter controls:

1. The userid charged by the CP accounting routines for the system resources used during a job.

2. The name and distribution code that appear on any spooled printer or punch output.

3. The userid to whom status messages are sent while the batch machine is executing the job.

Note: Items 1 and 2 are correct only if the directory for the userid involved contains the accounting option.

accntnum is your account number. This account number appears in the accounting data generated at the end of your job. It overrides the account number in the CP directory entry for the userid specified for this job.

jobname is an optional parameter that specifies the name of the job being run. If you specify a jobname, it appears as the CP spool file identification in the filetype field. The filename field always contains CMSBATCH. Refer to "Batch Facility Output" on page 212 for more information.

comments may be any additional information you want to provide.

The /* card indicates the end of a job to the batch facility. It takes the form:

```
/*
```

The batch facility treats all /* cards after the first as null cards. Therefore, if you want to ensure against the previous job not having a /* end-of-job indicator, you should precede your /JOB card with a /* card.

The /* card is also treated as an end-of-file indicator when a file is being read from the input stream. This is a special technique used in submitting source or data files through the card reader and is discussed under "A Batch EXEC for a Non-CMS User" on page 217.

Notes:

1. *Both "/JOB" and "/*" must begin in column 1.*
2. *The /* card can contain only the characters "/*." No other characters can appear on this input card.*

The /SET Card

The /SET card sets limits on a system's time, printing, and punching resources during the execution of a job. It takes the form:

```
/SET  [TIME seconds] [PRINT lines] [PUNCH cards]
```

where:

seconds	is a decimal value that specifies the maximum number of seconds of virtual CPU time a job can use.
lines	is a decimal value that specifies the maximum number of lines a job can print.
cards	is a decimal number that specifies the maximum number of cards a job can punch.

The default values for the batch facility are set at 32,767 seconds, printed lines, and punched cards per job. Any new limits defined using the /SET card must be less than these maximum settings. The system resources can be set at lesser values than the default values by an installation's system programmer; be sure you know the maximum installation values for batch resource limits before you use the /SET card.

A /SET card can appear anywhere in the job following the /JOB card. The new limits defined by the /SET card apply only to the part of the job that follows the /SET card.

A job can contain up to three /SET cards (one for each operand); a /SET card cannot be entered more than once for the same operand.

Only use /SET cards for the operands whose values you want to change from the default; the default values are reset between jobs. A /SET card for an operand overrides its default but does not reset the other operands.

Notes:

1. *Issuing the STIMER or TTIMER macros affects the CMSBATCH time limit. Depending upon the frequency, number and duration of STIMERs and/or TTIMERs issued, the CMSBATCH time limit may never expire. Blip processing is also affected under CMS BATCH and is not allowed.*

2. *"/SET" must begin in column 1.*

Other Input Records

The remainder of input records in the batch job consist of CP and CMS commands which will be executed. (For a description of command restrictions, refer to "Restrictions on CP and CMS Commands in Batch Jobs" on page 211). EXEC, EXEC 2, or System Product Interpreter statements cannot be imbedded in the input stream.

How the Batch Facility Works

The CMS batch facility, once initialized, runs continuously. When it begins executing a job, it sends a message to the userid of the user submitting the job. If you are logged on when the batch machine begins executing a job that you sent it, you receive the message:

```
MSG FROM BATCHID:  JOB 'yourjob' STARTED
```

When the batch machine finishes processing a job, it sends the message:

```
MSG FROM BATCHID:  JOB 'yourjob' ENDED
```

where yourjob is the jobname you specified on the /JOB card. Before it reads the next job from its card reader, the batch virtual machine:

- Closes all spooling devices and releases spool files
- Resets any spooling devices identified by the CP TAG command
- Detaches any disk devices that were accessed
- Punches accounting information to the system
- Reloads CMS.

All of this "housekeeping" is done by the CMS batch facility so that each job that is executed is unaffected by any previous jobs.

If a job that you sent to the batch virtual machine terminates abnormally (abends), the batch machine sends you a message:

```
MSG FROM BATCHID:  JOB 'yourjob' ABEND
```

and spools a CP storage dump of your virtual machine to the printer. The batch virtual machine stops processing your job and leaves the job in its reader in hold status.

Whenever the batch virtual machine has read and executed all of the jobs in its reader, it waits for more input.

Preparing Jobs for Batch Execution

When you want to submit a job to the CMS batch facility for execution, you should provide the same CMS and CP commands you would use to prepare to execute the same job in your own virtual machine.

You must provide the batch virtual machine with read access to any disk input files that are required for the job. You do this by supplying the LINK and ACCESS commands necessary. The batch virtual machine has an A-disk (195), so you can enter commands to access your disks as read-only extensions. For example, if you wanted the batch machine to execute a program module named LONDON on your 291 disk, your input file might contain the following:

```
/JOB FISH 012345
CP LINK MCGUIRE 291 291 RR SECRET
ACCESS 291 B/A
LONDON
```

Similarly, if you are using the batch virtual machine to execute a program using input and output files, you must supply the file definitions:

```
CP LINK ARDEN 391 391 RR FOREST
ACCESS 391 B/A
FILEDEF INFILE DISK VITAL STAT B
FILEDEF OUTFILE PUNCH
CP SPOOL PUNCH TO MCGUIRE
LONDON
```

If you expect printed or punched output from your job, you may need to include the spooling commands necessary to control the output. In the above example, the batch machine's punch is spooled to userid MCGUIRE's virtual reader.

Any output printer files produced by your job are spooled by the batch virtual machine to the printer. These files are spooled under your userid and with the distribution code associated with your userid, provided the userid's directory has the accounting option set. You can change the characteristics of these output files with the CP SPOOL command:

```
cp spool e class t
```

If you want output to appear under a name other than your userid, use the FOR operand of the SPOOL command:

```
cp spool e for jonson
```

Output punch files are spooled, by default, to the real system card punch (under your userid), unless you issue a SPOOL command in the batch job to control the virtual card punch of the batch virtual machine.

Restrictions on CP and CMS Commands in Batch Jobs

The batch facility permits the use of many CP and most CMS commands. The following CP commands can be used to control the batch virtual machine:

```
CHANGE          MSG
CLOSE           QUERY
DETACH          REWIND
DUMP            SMSG
DISPLAY         SPOOL
LINK            STORE
LOADVFCB        TAG
```

Notes:

1. *The CHANGE, CLOSE, and SPOOL commands may not be used to affect the virtual reader.*

2. *You cannot use the detach command to detach any spooling devices or the system or IPL disks.*

3. *The LINK command must be entered on one line in the format:*

   ```
   CP LINK userid vaddr vaddr mode password
   ```

   None of the LINK command keywords (AS, PASS, TO) are accepted. If the disk has no password associated with it, you must enter the password as ALL. A maximum of 26 links may be in effect at any one time.

4. *If a DIAGNOSE code X'08' is issued, the CHANGE and SPOOL commands will have an effect on the virtual card reader.*

All CP commands in a batch job must be prefaced with the "CP" command.

Note: If you are running a CMS, EXEC 2, or System Product Interpreter EXEC in the batch environment, the return code may be different than if the same EXEC is run standalone. BATCH intercepts some commands and checks their validity for the CMSBATCH environment. The return code may be from BATCH and not from CP.

Since the batch virtual machine reads input from its reader, you cannot use the following commands or operands that affect the reader:

```
ASSGN SYSxxx READER (CMS/DOS only)
DISK LOAD
FILEDEF READER
READCARD
RECEIVE
```

The RDCARD macro cannot be used by jobs that run under the CMS batch machine.

Invalid SET command operands are:

```
BLIP        PROTECT
IMPCP       REDTYPE
INPUT       RELPAGE
OUTPUT
```

All of the other CMS SET command operands can be used in a job executing in the batch virtual machine. All forms of the CP SET command are invalid.

Note: If the SET TIMER REAL command is used for the batch machine, the timer expires every two seconds (including while batch is waiting for the reader). To avoid this problem, use the command SET TIMER ON.

Batch Facility Output

Any files that you request to have printed during your job's execution are spooled to the real system printer under your userid, unless you have spooled it otherwise. Once released for processing, these output files are under the control of the CP spooling facilities; if you are logged on, you can control the disposition of these files before they are printed with the CLOSE, PURGE, ORDER, and CHANGE commands. See the following section "Purging and Reordering Batch Jobs."

Output files produced by the batch virtual machine are identifiable by the filename CMSBATCH in the CP spool file name field. The spool file type field contains the filetype JOB, unless you specified a jobname on the /JOB card. This applies to both printer and punch output files.

In addition to your regular printed output, the CMS batch facility spools a console sheet that contains a record of all the lines read in, and the responses, error messages, and return codes that resulted from command or program execution. This file is identified by a spool file name of BATCH and a spool file type of CONSOLE.

Purging and Reordering Batch Jobs

If you are logged on to the batch userid, you can control the execution of batch virtual machine jobs when required by purging, reordering, and restarting them; by the same token, because all the closed printer files are queued for system output under the submitting userid, you can change, purge, or reorder these files prior to processing on the system printer.

To purge a job executing under the batch monitor, follow the procedure below:

1. Signal attention and enter the virtual machine environment.
2. Enter the HX (halt execution) Immediate command.
3. Disconnect the virtual machine using the CP DISCONN command.

The HX command causes the batch facility to abnormally terminate. This provides the user with an error message and a CP dump of the batch facility virtual machine. The batch monitor then loads itself again and starts the next job (if any).

To purge an individual input spool file that is not yet executing, issue the CP PURGE command:

```
PURGE READER spoolid
```

In the format above, spoolid is the spool file number of the job to be purged from the batch virtual machine's job queue. For example, the statement:

```
purge reader 123
```

would purge 123 from the batch virtual machine's job queue.

To reorder individual spool files in the batch facility's job queue, use the CP ORDER command:

```
ORDER READER spoolid1 spoolid2...
```

In this format, spoolid1 and spoolid2 are the assigned spool file identifications of the jobs to be reordered.

You can determine which jobs are in the queue by using the CP QUERY command:

```
query reader all
```

This QUERY command lists the filenames and filetypes of all the jobs in the batch virtual machine's job queue. You can then reorder them, using the ORDER command.

Using EXEC Files for Input to the Batch Facility

There are a variety of ways that EXEC procedures can help facilitate the submission of jobs to the CMS batch facility. You can prepare an EXEC file that contains all of the CMS commands you want to execute, and then pass the name of the EXEC to the batch virtual machine. For example, consider the files COPY JCL and COPYF EXEC:

```
COPY JCL:    /JOB CARBON 999999
             EXEC COPYF
             /*

COPYF EXEC:  COPYFILE FIRST FILE A SECOND = =
             COPYFILE THIRD FILE A FOURTH = =
```

Then, if you enter the commands:

```
spool punch to cmsbatch
punch copy jcl * (noheader
```

the commands in the EXEC file are executed by the batch virtual machine.

You could also use an EXEC to punch input to the batch virtual machine. Using the same commands as in the example above, you might have an EXEC named BATCOPY:

```
/* EXEC to submit a batch job */
'CP SPOOL PUNCH TO BATCH3'
punch = execio 1 punch '('string
punch '/JOB CARBON 999999'
punch 'COPYFILE FIRST FILE A SECOND = ='
punch 'COPYFILE THIRD FILE A FOURTH = ='
punch '/*'
'CP CLOSE PUNCH'
```

Then, when you enter the EXEC name:

```
batcopy
```

the input lines are punched to the batch virtual machine.

The examples above are very simple; you probably would not go to the trouble of sending such a job to the batch virtual machine for processing. The examples do, however, illustrate the two basic ways that you can use EXEC procedures with the batch facility:

1. Invoking an EXEC procedure from a batch virtual machine

2. Using an EXEC procedure to create a job stream for the batch virtual machine.

In either case, the EXECs that you use may be very simple or very complicated. In the first instance, an EXEC might contain many steps, with control statements to conditionally control execution, error routines, and so on.

In the second instance, you might have an EXEC that is versatile so that it can be invoked with different arguments so as to satisfy more than one situation. For example, if you want to create a simple EXEC to send jobs to the batch virtual machine to be assembled, it might contain:

```
/* An EXEC for batch assemblies */
'CP SPOOL PUNCH TO BATCH3 CONT'
arg filename .
punch = execio 1 punch '('string
punch '/JOB MCGUIRE 888888'
punch 'CP LINK MCGUIRE 191 391 RR LINKPASS'
punch 'ACCESS 391 B/A'
punch 'ASSEMBLE' filename '(PRINT'
punch 'CP SPOOL PUNCH TO MCGUIRE'
punch 'PUNCH' filename 'TEXT A (NOHEADER'
punch '/*'
'CP SPOOL PUNCH NOCONT CLOSE'
```

If this file were named BATCHASM EXEC, then whenever you wanted the CMS batch facility to assemble a source file for you, you would enter:

```
batchasm filename
```

and the batch virtual machine would assemble the source file, print the listing, and send you a copy of the resulting TEXT file.

Sample System Procedures for Batch Execution

To extend the above example a little further, suppose you wanted to process source files in languages other than the assembler language. You want, also, for any user to be able to use this EXEC. You might have a separate EXEC file for each language, and an EXEC to control the submission of the job. This example shows the controlling EXEC file BATCH and the ASSEMBLE EXEC.

BATCH EXEC

```
/*
 *   This EXEC submits assemblies/compilations to CMS Batch
 *
 *   - Punch batch job card;
 *   - Call appropriate language EXEC to punch executable commands
 */
    arg userid filename language
    if language = ''
      then do
        say 'Correct form is: BATCH userid fname ftype (language)'
        exit 100
      end
    punch = 'EXECIO 1 PUNCH (STRING'
    trace errors
    signal on error
    cp spool d cont to batchcms
    punch '/JOB' userid '1111' filename
    punch 'CP LINK' userid '191 291 RR SECRET'
    punch 'ACCESS 291 B/A'
    exec language filename userid
    punch '/*'
    cp spool d nocont
    cp close d
    cp spool d off
    exit
    Error: exit 100
```

ASSEMBLE EXEC

```
/*
 *   Correct form is : ASSEMBLE fname userid
 *
 *   Punch commands to:
 *     - Invoke CMS assembler
 *     - Return text deck to caller
 */
    arg fname userid
    signal on error
    trace errors
    punch = 'EXECIO 1 PUNCH (STRING'
    punch 'GLOBAL MACLIB UPLIB CMSLIB OSMACRO'
    punch 'CP MSG' userid 'Asmbling' fname
    punch 'ASSEMBLE' fname '(PRINT NOTERM)'
    punch 'CP MSG' userid 'Assembly done'
    punch 'CP SPOOL D TO' userid 'NOCONT'
    punch 'PUNCH' fname 'TEXT A1 (NOHEADER)'
    punch 'CP CLOSE D'
    punch 'CP SPOOL D OFF'
    punch 'RELEASE 291'
    punch 'CP DETACH 291'
    exit
    Error: exit 102
```

Executing the Sample EXEC Procedure

If the above EXEC procedure is invoked with the line:

```
batch fay payroll assemble
```

the BATCHCMS virtual machine's reader should contain the following statements (in the same general form as a FIFO console stack):

```
/JOB FAY 1111 PAYROLL
CP LINK FAY 191 291 RR SECRET
ACCESS 291 B/B
GLOBAL MACLIB UPLIB CMSLIB OSMACRO
CP MSG FAY Asmbling PAYROLL
ASSEMBLE PAYROLL (PRINT NOTERM)
CP MSG FAY Assembly done
CP SPOOL D TO FAY NOCONT
PUNCH PAYROLL TEXT A1 (NOHEADER)
CP CLOSE D
CP SPOOL D OFF
RELEASE 291
CP DETACH 291
/*
```

When the batch facility executes this job, the commands are executed as you see them: if you are logged on, you receive, in addition to the normal messages that the batch facility issues, those messages that are included in the EXEC.

A Batch EXEC for a Non-CMS User

Many installations run the CMS batch facility for non-CMS users to submit particular types of jobs. Usually, a series of EXEC files are stored on the system disk so that each user only needs include a card to invoke the EXEC, which executes the correct CMS commands to process data included with the job stream.

For example, if a non-CMS user wanted to compile FORTRAN source files, the following BATFORT EXEC file could be stored on the system disk:

```
/* EXEC for batch FORTRAN Compiles */
arg filename
'FILEDEF INMOVE TERM (RECFM F BLOCK 80 LRECL 80'
'FILEDEF OUTMOVE DISK' filename 'FORTRAN A1 (RECFM F LRECL 80 BLOCK 80'
'MOVEFILE'
'GLOBAL TXTLIB FORTRAN'
'FORTGI' filename '(PRINT)'
fortret = rc
if rc = 0 then
  'PUNCH' filename 'TEXT A1 (NOHEADER)'
exit fortret
```

To use this EXEC, a non-CMS user might place the following real card deck in the system card reader:

```
ID CMSBATCH
/JOB  JOEUSER 1234 JOB10
BATFORT  JOEFORT
.
.
.
source file
.
.
.
/* (end-of-file indicator)
/* (end-of-job indicator)
```

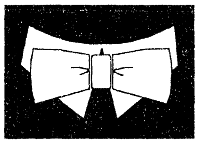
When the batch virtual machine executes this job, it begins reading the EXEC procedure from disk, and executes one line at a time. When it encounters the MOVEFILE command, it begins reading the source file from its card reader (the batch facility interprets a terminal read as a request to read from the card reader). It continues reading until it reaches the end-of-file indicator (the /* card), and then resumes processing the EXEC on the next line following the MOVEFILE command.

Additional functions may be added to this EXEC procedure, or others may be written and stored on the system disk to provide, for example, a compile, load, and execute facility. These EXEC procedures would allow an installation to accommodate the non-CMS users and maintain common user procedures.

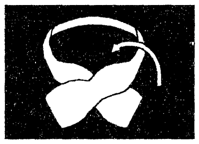
Part 3: Learning to use EXECs

The CMS facilities known as the System Product Interpreter, EXEC 2 and CMS EXEC processors or interpreters allow you to create EXEC files. Using EXEC files, you can execute many commands and programs by entering a single command from your terminal; in effect, this is like writing your own CMS commands.

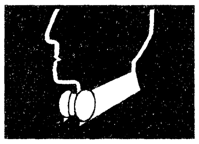
In this part, the EXEC facilities are described in general terms to acquaint you with the expressions used in EXEC files and the basic way that EXECs function.



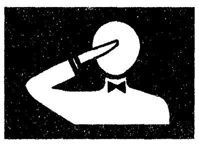
Chapter 11: Introduction to the EXEC Processors presents a survey of the basic characteristics and functions of EXEC facilities available to you.



Chapter 12: Creating System Product Interpreter EXECs describes how to create and invoke System Product Interpreter EXECs. Sample EXECs are provided for you to try.



Chapter 13: Creating a PROFILE EXEC describes how you can create your own PROFILE EXEC.



Chapter 14: CMS Commands Used Along With EXECs provides examples of using some CMS commands with System Product Interpreter EXECs.

Chapter 11. Introduction to the EXEC Processors

Three EXEC processors are available:

- System Product Interpreter
- EXEC 2
- CMS EXEC.

The System Product Interpreter handles System Product Interpreter programs, which are written in the Restructured Extended Executor (REXX) language. The EXEC 2 processor handles EXEC 2 programs. The CMS EXEC processor handles CMS EXEC programs. EXEC 2 programs and processing are similar to those of the CMS EXEC. The System Product Interpreter programs are *not* similar to those of EXEC 2 or CMS EXEC.

The System Product Interpreter

The System Product Interpreter is an interpretive command and macro processor. It coexists with the CMS EXEC and EXEC 2 processors. The System Product Interpreter is functionally a superset of CMS EXEC and EXEC 2, but it uses a completely different language and syntax. There is no compatibility between System Product Interpreter programs and those of CMS EXEC or EXEC 2.

VM/SP differentiates System Product Interpreter programs from CMS EXEC or EXEC 2 programs by their first statement. The first statement of every System Product Interpreter program must be a comment. A comment begins with a /*, and ends with an */, with anything you want in between. For example:

```
/* This is a comment. */
```

The System Product Interpreter functions are easy to learn and use. They use a general-purpose, high-level language called REXX, much like that used by PL/I and other high-level programming languages. REXX instructions use structured programming concepts like IF/THEN/ELSE, SELECT, DO WHILE, etc, which allow you to write programs while using words much like those you use to think and communicate.

Other features of the System Product Interpreter and the REXX language are:

- It has a number of useful built-in functions you can use in your programs.

- Programs may be written in mixed case with free form layout (which makes them easier to read and follow).

- It has extensive mathematical capabilities (you can even use it as a desk calculator if you wish).

- There is no limit (except the user's virtual storage size) to the length of manipulated data.

- It is easy to find syntax errors in a program. The System Product Interpreter executes programs line-by-line and word-by-word, without translating them to another form (no compiling). Thus, when there's a syntax error, the place where it occurred is clearly indicated.

- You can use the TRACE instruction to see how the System Product Interpreter is interpreting a particular instruction. This should help you in debugging.

The following books tell you how to use the System Product Interpreter and the REXX language:

- The *VM/SP System Product Interpreter User's Guide* is a step-by-step, tutorial-like, guide to using the System Product Interpreter. It is intended for new users. There is also an introductory chapter in the *VM/SP CMS Primer* about the System Product Interpreter.

- The *VM/SP System Product Interpreter Reference* is a complete compilation of reference information for using the System Product Interpreter. It is intended for all users.

As a CMS user, you should become familiar with the System Product Interpreter and use it often to tailor CMS commands to your own needs, as well as to create your own commands. The System Product Interpreter and the EXEC 2 interpreter may be used by the System Product Editor for XEDIT macro processing support.

Complete details about using the System Product Interpreter can be found in the books listed above.

The EXEC 2 Processor

The EXEC 2 processor handles EXEC 2 programs. These EXEC 2 programs and processing are similar to CMS EXEC programs and processing.

EXEC 2 differs from CMS EXEC in the following ways:

- There is no 8-byte token restriction. Statements are composed of "words" of up to 255 characters each.
- Commands may be issued from EXEC 2 either to CMS or to specified "subcommand" environments, for example the System Product Editor.
- EXEC 2 has extended string manipulation functions.
- EXEC 2 has arithmetic functions for multiplication and division.
- EXEC 2 has extended debugging facilities.
- EXEC 2 supports user defined functions and subroutines.
- EXEC 2 allows CMS user programs to manipulate EXEC 2 variables.

The CMS EXEC Processor

A CMS EXEC procedure is a CMS file that contains executable statements. The statements may be CMS or CP commands or EXEC control statements. The execution can be conditionally controlled with additional EXEC statements, or it may contain no EXEC statements at all. In its simplest form, an EXEC file may contain only one record, have no variables, and expect no arguments to be passed to it. In its most complex form, it can contain thousands of records and may resemble a program written in a high-level programming language.

Two CMS commands create EXEC files. One is LISTFILE, which can be invoked with the EXEC option; it creates a file named CMS EXEC. The CMS/DOS command, LISTIO, creates an EXEC file named $LISTIO EXEC, which creates records for each of the system and programmer logical unit assignments. For details on the LISTIO command and the $LISTIO EXEC, refer to the *VM/SP CMS for System Programming.*

The EXEC control statements for the CMS EXEC facility are described in the *VM/SP CMS Command Reference.*

Relationship of the EXEC interpreters

The three interpreters described above have their own distinct keywords and syntax. So for example, you may not place EXEC 2 statements within a System Product Interpreter program.

The three interpreters coexist, so EXEC programs will continue to execute correctly with no user modifications, regardless of the language. To run CMS EXEC programs as EXEC 2 programs, you must convert the EXEC programs to EXEC 2 programs.

While you may not use, for example, EXEC 2 language statements in an EXEC to be interpreted by the System Product Interpreter nor REXX language statements in an EXEC to be interpreted by the EXEC 2 interpreter, any EXEC may call another EXEC, regardless of the language. Thus an EXEC 2 procedure may be invoked from within a CMS EXEC procedure, and vice versa.

To allow greater user flexibility with EXEC 2 and the System Product Interpreter, automatic cleanup of an active OS, VSAM, or Vector environment is not invoked at command completion as it is in the CMS EXEC processor. It is your responsibility to ensure that OS, VSAM, and Vector cleanup functions are invoked when needed. VSAM cleanup can be invoked explicitly by issuing 'DMSVSR' as a command. The CMS EXEC processor invokes OS, VSAM, and Vector cleanup after the execution of any CMS command. Consequently, any CMS EXEC invoked resets the OS, VSAM, and Vector environments if it contains a CMS command that is executed.

Invoking EXECs

EXEC programs may reside in EXEC files (with a filetype of EXEC) on disk or in storage as storage resident EXECS, and can be invoked via the 'EXEC' command.

When an EXEC file is invoked, CMS examines the first statement of the EXEC file to determine which EXEC processor must handle it. If the first line contains /* a comment */, then the System Product Interpreter is called. If the first statement of the EXEC is &TRACE, CMS calls the EXEC 2 processor to handle it. If the first statement is not &TRACE or /* a comment */, CMS calls the EXEC processor to handle it.

Note: The &TRACE statement does not have to be the first statement in an EXEC 2 file if the file does not have a filetype of EXEC (if the EXEC is invoked by an SVC 202).

Attributes of EXEC Files

EXEC files can have any filename that is valid for a CMS filename. EXEC 2 and System Product Interpreter files have filetype EXEC for files that are invoked from the CMS environment, and the filetype XEDIT for files used as System Product editor macros.

System Product Interpreter or EXEC 2 files can be either "F" or "V" format. The maximum line length for lines read from the console is 130; for lines read from the stack it is 255.

Chapter 12. Creating System Product Interpreter EXECs

Creating a System Product Interpreter EXEC

A System Product Interpreter file, like a CMS EXEC or EXEC 2 file, has a filetype of EXEC. To determine which EXEC interpreter will be invoked by an EXEC file, look at the first line in the file.

First line	Interpreter
/* a comment */	System Product Interpreter
&TRACE	EXEC 2 Processor
Anything else.	CMS EXEC Facility

You can create EXEC files with the CMS editors, by punching cards, or by using CMS commands or programs. When you create a file (filetype of EXEC) using XEDIT, records are, by default, variable-length with a logical record length (lrecl) of 130 characters and case is upper. The CMS EXEC facility can process variable-length files of up to 130 characters. EXEC 2 can process variable-length files of up to 255 characters. The System Product Interpreter processes files of any logical record length (lrecl). For example, to create an EXEC file, enter:

```
xedit new exec
```

If you have a fixed-length file that you want to convert to a variable-length file, then you can edit the EXEC file and issue the XEDIT subcommand:

```
recfm v
```

Or, you can use the COPYFILE command:

```
copyfile new exec a (recfm v
```

Whenever possible, you should use variable-length EXEC files.

If you use XEDIT to create a CMS EXEC or an EXEC 2 EXEC, you cannot enter the EXEC statements in mixed case. Use the XEDIT subcommand:

```
set case uppercase
```

Invoking Your EXEC Files

EXEC procedures are invoked when you enter the filename of the EXEC file. You can precede the filename on the command line with the CMS command, EXEC. For example:

```
exec test
```

where TEST is the filename of the EXEC file. For example, an EXEC named THANKYOU would be executed when you entered either:

```
exec thankyou
 -- or --
thankyou
```

You must precede the EXEC filename with the EXEC command when:

- You invoke an EXEC from CMS EXECs and EXEC 2 EXECs.
- You invoke an EXEC from REXX with "address command." (The default is "address CMS," which means EXEC need not be specified.)
- You invoke an EXEC from a program.
- You call a System Product Interpreter EXEC recursively.
- You have the implied EXEC (IMPEX) function set OFF for your virtual machine.

The implied EXEC (IMPEX) function is controlled by the SET command. It allows you to treat EXEC files as commands so that you only must enter the filename of the EXEC program. The default setting for IMPEX is ON; you almost never need to change it. To find out what the IMPEX setting is, enter:

```
q impex
```

If the response is:

```
IMPEX    = OFF
```

this means that the EXEC command must precede the EXEC filename to invoke an EXEC procedure. To set IMPEX to ON, so that you only need to enter the EXEC filename, enter:

```
set impex on
```

An EXEC procedure having a synonym defined for it can be invoked by its synonym if the implied EXEC (IMPEX) function is on. You may use the synonym for an EXEC program within a System Product Interpreter program.

One EXEC file that you never have to specifically invoke is a PROFILE EXEC. It automatically executes after you IPL CMS, when your A-disk is accessed. PROFILE EXECs are discussed in Chapter 13, "Creating a PROFILE EXEC."

Sample System Product Interpreter EXECs

Here are two sample System Product Interpreter EXECs to give you some flavor of the language.

The first sample is an EXEC to copy a file from any disk to the user's A-disk. Note that the EXEC uses the required first comment statement as a description of its function.

```
/* This exec copies a file from any disk to the user's A-disk */
arg fn ft fm extra
if fn = '?' then signal tell
if extra ¬= '' | ft = ''
then do
   Parse source . . me .
   say 'Invalid command for 'me' exec.'
   exit
   end
if fm  = '' then fm = '*'
copyfile fn ft fm '= = a'
exit rc
tell:
parse source . . me .
say 'This exec,' me', copies the given file to'
say 'the A-disk and passes back the return'
say 'code from copyfile'.
exit 100
```

The second sample sends the file that you specify to the userid that you specify. Note that in System Product Interpreter EXECs you do not need to preface a CP command with CP.

```
/* This exec sends a specific file to a specific user */
parse source . . me .
arg user fn ft fm extra
if fn = ''
  then
    do
      say 'Command is:' me 'user fn ft <fm>'
      exit 100
    end
if ft = '' | extra ¬= ''
  then
    do
      say 'Invalid' me 'message'
      exit 101
    end
spool punch to user class a
if  rc ¬= 0
  then
    do
      say user 'is not a valid userid'
      exit 102
    end
if fm = '' then fm = 'A'
punch fn ft fm
retsave = rc
spool punch to '*' class a
if retsave ¬= 0
  then
    do
      say 'Error' retsave 'from punch (while in' me')'
      exit 103
    end
msg user 'I have punched you my file' fn ft fm
exit
```

Complete details about the System Product Interpreter can be found in the *VM/SP System Product Interpreter User's Guide* and in the *VM/SP System Product Interpreter Reference.*

Chapter 13. Creating a PROFILE EXEC

A PROFILE EXEC is different from other EXECs. It has the special filename PROFILE and it is executed automatically whenever you issue "IPL CMS" (or if you have automatic IPL). Your PROFILE EXEC contains the CP and CMS commands that you issue at the start of every terminal session. You can write your PROFILE EXEC for any of the EXEC interpreters. It usually contains commands that:

- Access disks
- Describe your terminal and printer
- Set up your PF keys
- Describe macro and text libraries that you commonly use
- Set your screen colors (color terminals only)
- Invoke your synonym table
- Make frequently used EXECs storage resident.

A PROFILE EXEC written with System Product Interpreter statements might look like this:

```
/* sample profile */
access 497 b                  /* Access B-disk               */
set rdymsg smsg               /* Short form of ready msg     */
set blip '*'                  /* set blip character to *     */
synonym mysyn                 /* Invoke my synonym table     */
global maclib osmacro privmac  /* MACRO libraries             */
global txtlib privlib         /* TEXT libraries              */
set PF1 immed rdrlist         /* PF1 key set to RDRLIST      */
set PF6 retrieve             /* PF6 key RETRIEVE function */
set PF11 immed filelist       /* PF11 key set to FILELIST    */
execload filelist exec '('system /* Make FILELIST, RECEIVE,    */
execload receive exec '('system  /* and EXECUTE storage        */
execload execute xedit '('system /* resident EXECs.            */
```

Do not use the CP DEFINE STORAGE command in your PROFILE EXEC. It resets your virtual machine and you would have to IPL CMS again.

You can enter "profile" at any time to execute your PROFILE EXEC. If you make changes to your PROFILE EXEC during your terminal session, the changes will not be in effect until you execute your profile again.

Should you want to suppress the execution of your PROFILE EXEC, the first command you enter after you issue the IPL command is the CMS ACCESS command with the NOPROF option specified. For example, if you enter:

```
ipl cms
```

The system response may be:

```
VM/SP Release 5  10/31/86  11:22:33
```

To suppress the execution of your PROFILE EXEC, you enter:

```
acc (noprof
```

When the system responds with:

```
Ready;
```

you have loaded CMS and accessed your A-disk without invoking your PROFILE EXEC.

You can find more information about the CMS ACCESS command in the *VM/SP CMS Command Reference.*

The EXECLOAD command is used to make a particular EXEC storage resident. The EXEC remains storage resident for the entire session and consequently, does not need to be reloaded each time it is invoked. For example, if you are a frequent user of the FILELIST, MACLIST, NOTE, NAMES, RDRLIST, SENDFILE, and TELL commands, you might consider making the following EXECs storage resident:

DISCARD	EXEC
EXECUTE	XEDIT
FILELIST	EXEC
MACLIST	EXEC
NAMES	EXEC
NOTE	EXEC
PEEK	EXEC
PROFFLST	XEDIT
PROFMLST	XEDIT
PROFNOTE	XEDIT
PROFPEEK	XEDIT
PROFRLST	XEDIT
PROFSEND	XEDIT
RDRLIST	EXEC
RECEIVE	EXEC
RECEIVE	XEDIT
SENDFILE	EXEC
TELL	EXEC
X$FLST$X	XEDIT
X$MLST$X	XEDIT
X$NAME$X	XEDIT
X$PEEK$X	XEDIT
X$SEND$X	XEDIT

If your installation has an Installation Discontiguous Shared Segment (DCSS) containing any of these EXECs, you would not want to load them into storage. Frequently used EXECs are loaded into a DCSS so that users can share the same executing copy of the EXEC. To use the EXECs in an Installation DCSS, you can load the shared segment when you IPL CMS, or you can issue SET INSTSEG ON.

To find more information about using the Installation DCSS, see the IPL command in the *VM/SP CP Command Reference* and the SET, QUERY, and EXECMAP commands in the *VM/SP CMS Command Reference*.

13

Chapter 14. Commands Used with System Product Interpreter EXECs

The following commands are used along with System Product Interpreter EXECs. Command formats, descriptions and usage notes for these commands are found in the *VM/SP CMS Command Reference.*

EXECDROP	Removes EXECs and Macros from storage or discontinues use of EXECs and Macros in an Installation Discontiguous Shared Segment (DCSS).
EXECIO	Manages movement of lines between virtual devices and the program stack or a variable. Also causes execution of CP commands and recovers resulting output.
EXECLOAD	Loads an EXEC into storage.
EXECMAP	Lists EXECs and Macros in storage and in the Installation DCSS.
EXECOS	Resets the OS, VSAM, and Vector environments under CMS without returning to the interactive environment.
EXECSTAT	Provides the status of a specified EXEC.
GLOBALV	Sets, maintains, and retrieves a collection of named variables.
IDENTIFY	Displays or stacks userid, nodeid, rscsid, date, time, time zone, and day of the week.
IMMCMD	Establishes or cancels Immediate commands from an EXEC.
LISTFILE	Lists information about CMS disk files.
NAMEFIND	Displays/stacks information from a NAMES file (default 'userid NAMES').

QUERY	Requests information about a CMS virtual machine.
RDR	Generates a return code and either displays or stacks a message that identifies the characteristics of the next file in your virtual reader.
SET EXECTRAC	Sets tracing ON or OFF for your System Product Interpreter or EXEC 2 EXEC.
SET INSTSEG	Sets the access to the Installation DCSS to ON or OFF and sets the mode where it is searched.
WAITREAD VSCREEN	Use the WAITREAD VSCREEN command from an EXEC to update the virtual screen with data in the virtual screen queue, refresh the physical screen, and wait for the next attention interrupt.
XEDIT	Invokes the System Product Editor to create or modify a disk file.

The following Immediate commands can be used along with System Product interpreter EXECs:

HI	halt interpretation
TS	trace start
TE	trace end

The following examples show you how you may use some of the CMS commands with your System Product Interpreter EXECs:

Using EXECIO

The EXECIO command manages movement of lines between virtual devices and the program stack. It also causes execution of CP commands and recovers resulting output.

This example is not intended to teach you all you need to know to write System Product Interpreter programs. If you are not already familiar with the System Product Interpreter, see the *VM/SP System Product Interpreter User's Guide.*

The example illustrates how you might use EXECIO commands in a System Product Interpreter program to read a CMS file from the program stack, then print that file, 60 lines per page, with the output indented 15 spaces.

This is not the only, nor necessarily the best, way to accomplish the results. However, it does show some uses of the EXECIO command within a System Product Interpreter program. The statement numbers in the left margin are to reference explanations below, and are not a part of the program.

Because the program reads, prints, and indents, let's name it RDPRIND EXEC (the filetype must be EXEC).

```
RDPRIND EXEC

1. /* This program reads, prints, and indents */
2. trace n
3. execio 1 print '('cc 1 string
4. arg filename filetype filemode .
5.    do until execiorc ¬=0
6.    execio 100 diskr filename filetype filemode stem line.
7.    execiorc=rc
8.       do I=1 to line.0
9.       execio 1 print '('string'                  'line.i
10.        if I//60=0
11.           then execio 1 print '('cc 1 string
12.        end
13.     end
14. close prt name filename filetype filemode
15. exit
```

The following explains the meaning of each statement in the RDPRIND program:

1. The first statement in a System Product Interpreter program must always be a comment (/* comment */). Note how we use the comment to tell what the program does.

2. Trace all host commands which return a negative return code.

3. This is a CMS command to write a line to the printer (space to top of new page).

4. Read in the passed parameters, assigning values to filename, filetype, and filemode. The "." is a placeholder, used here to ignore any passed data after the third parameter.

5. Starts a DO loop. This statement says that the instructions following the DO, up to the END statement which is paired with DO (line 13), should be repeated until the return code from EXECIO (saved in EXECIORC) is not 0.

6. This is a CMS command to read 100 lines from the file called "filename filetype filemode". Those values are set by the ARG command in line 4. The number of lines actually read is assigned to variable LINE.0 and the actual file lines are assigned to the variables LINE.1, LINE.2, etc.

7. The return code from the previous host command (in this case from EXECIO on line 6) is saved in the special variable named RC. This

statement saves the return code in a variable called EXECIORC so it can be checked later.

8. Another DO loop starts here, similar to the one started in line 5. In this loop, the set of instructions between the DO and its END (on line 12) will be repeated while I is incremented from 1 until it is equal to the number of lines returned from EXECIO.

9. This is a CMS command to write a line to the printer. The blanks will be preserved and the value of LINE.I will be placed on the end of the command before it is passed to CMS.

10. This is a conditional check. It asks if the remainder of I divided by 60 is equal to 0. This will be true when I=60, 120, etc.

11. If the previous condition checked (in line 10) is true, then this line is executed. If it's executed, it spaces the printer to the top of a new page (the same command was used in line 3).

12. This END ends the DO loop started in line 8.

13. This END ends the DO loop started in line 5.

14. This is a CP command to close the printer and name the file. Its filename, filetype, and filemode will be set based on the values set in line 4.

15. This statement ends normal processing.

Now, to cause the EXEC to read and print a CMS disk file named TESTFILE DATA A, issue:

```
RDPRIND TESTFILE DATA A
```

TESTFILE, DATA, and A are substituted into the program for filename, filetype, and filemode respectively.

Using EXECDROP, EXECLOAD, EXECMAP, and EXECSTAT

The EXECLOAD command loads an EXEC into storage and prepares the EXEC for execution. The following command:

```
execload tphone exec a
```

loads TPHONE EXEC into user free storage. When the EXEC is subsequently invoked, the storage resident EXEC is executed. This eliminates the need for CMS to reload the EXEC into storage each time the EXEC is invoked. The EXEC remains storage resident during the entire session or until specifically dropped by the EXECDROP command. Be aware that if you make any changes to the EXEC file on disk, the storage resident copy of the EXEC remains unchanged. To have the new version execute, you will have to do one of the following:

- Drop the EXEC from storage (using EXECDROP).
- Drop the EXEC from storage (using EXECDROP) and reload it (using EXECLOAD).
- Load the new version (EXECLOAD with the PUSH option).

Note: Before loading an EXEC into storage, you should determine whether or not you are using the Installation Discontiguous Shared Segment (DCSS) and if an EXEC with the same file identifier resides in it. If so, to load the EXEC you should:

- Specify the PUSH option on the EXECLOAD command,
- Use the EXECDROP command to drop your access to that EXEC and then issue the EXECLOAD command, or
- Use the SET INSTSEG OFF command to discontinue use of the Installation DCSS and then issue the EXECLOAD command.

To verify the existence of the TPHONE EXEC in storage and on disk, you can use the EXECSTAT command. For example:

```
execstat tphone exec
```

gives a return code of 0 in register 15, verifying that the EXEC is storage-resident. The EXECMAP command lists the storage-resident EXECs.

```
execmap
```

will return the following if TPHONE EXEC is the only storage-resident EXEC:

```
Name      Type      Usage     Records     Bytes     Attribute
TPHONE    EXEC          0          15       512     USER
```

Should you decide that you no longer require an EXEC to be storage resident, you can delete it from storage with the EXECDROP command. For example, issuing:

```
execdrop tphone exec
```

will delete the TPHONE EXEC from storage.

Using IPL, SET INSTSEG, EXECMAP, and EXECDROP

The IPL command links the Installation Discontiguous Shared Segment (DCSS) for your CMS session. The command:

```
ipl cms parm instseg yes
```

links the default Installation DCSS, CMSINST. The DCSS contains the executing copy of frequently used EXECs, eliminating the need to load your own copy of the EXECs into storage. The system accesses the DCSS at mode S, meaning the DCSS will be searched before the S-disk (system disk).

To change the access mode, use the SET INSTSEG command. For example, type:

```
set instseg on b
```

to access the Installation DCSS at mode B. If you already have a B-disk, then the DCSS is searched immediately before it.

The EXECMAP command lists the EXECs in storage and the ones in the DCSS. EXECMAP returns a list with the Attribute specifying the location of the EXEC. For example, if you had loaded TPHONE EXEC into user storage, NAMES EXEC into system storage, and the FILELIST EXEC and RECEIVE EXEC were in the Installation DCSS, the list would be:

```
Name      Type      Usage  Records  Bytes  Attribute
TPHONE    EXEC          0       15    515  USER
NAMES     EXEC          0       60   4616  SYSTEM
FILELIST  EXEC          0      163   7488  SHARED
RECEIVE   EXEC          0      625  27568  SHARED
```

If you no longer want to use an EXEC in the DCSS, you can discontinue your access to it with the EXECDROP command. For example, issuing:

```
execdrop receive exec (shared
```

deletes your access to the RECEIVE EXEC in the DCSS. If you decide you want to drop your access to the DCSS entirely, use the SET INSTSEG command. SET INSTSEG OFF discontinues use of the Installation DCSS until you set it ON or until you re-IPL CMS.

Using EXECOS

The EXECOS command resets the OS, VSAM, and Vector environments under CMS without returning to the interactive environment. If you request a reset of the OS, VSAM, or Vector environment, after the execution of a CMS EXEC, the EXECOS command should *precede* the CMS EXEC command. For example:

```
/* example of using EXECOS within an EXEC */
execos exec vmfasm dmsseb dmssp
exit
```

Using GLOBALV

The GLOBALV command sets, maintains, and retrieves a collection of named variables. You can pass these global variables between EXECs.

For example, we have two EXEC files named FIRST EXEC and SECOND EXEC, where the FIRST EXEC calls the SECOND EXEC. The variables are established as global variables in the SECOND EXEC by the statement "globalv put RUMORS." The statement "globalv get RUMORS" in the FIRST EXEC assigns the global variables to the FIRST EXEC.

```
/* first exec */
 .
 .
 .
 second
 .
 globalv get 'RUMORS'
 .
 .
 .
 exit
```

```
/* second exec */
 .
 .
 .
 .
 globalv put 'RUMORS'  /* assign variables */
 .
 .
 .
 exit
```

Using IDENTIFY

You can use the information returned by the IDENTIFY command within your EXEC.

For example:

```
/* example of using identify within your exec */
.
.
.
'identify(lifo'   /* get some useful information */
pull userid . node . rscsid .
.
.
.
exit
```

Using IMMCMD

The IMMCMD command establishes or cancels Immediate commands from an EXEC.

For the following example, we will assume that you have an EXEC that performs a repetitive process. Each time this EXEC is processed, one record is logged to disk. Suppose you wanted to suppress logging of the disk records without terminating the EXEC. Since HX terminates the EXEC, you would not want to use it. Using Pull is not a good alternative since you want to decide at what point to terminate disk logging. You can create your own Immediate command to stop disk logging using the CMS IMMCMD command within your EXEC. For example:

```
/*  Sample EXEC using the CMS IMMCMD command          */
/* Set up stoplog Immediate command                  */
IMMCMD SET STOPLOG
/* Set default logging                               */
arg log .
if log='' then log='YES'
if log¬='YES' & log¬='NO' then do
   say 'Invalid parameter :' log
   exit 24
   end
do forever
   /* Check for STOPLOG */
   IMMCMD STATUS STOPLOG
   if rc¬=0 then log='NO'
   /* Perform process ... */
        .
        .
   if log='YES' then EXECIO 1 DISKW LOG FILE A
        .
        .
   end
/* Clear STOPLOG Immediate command */
IMMCMD CLEAR STOPLOG
exit
```

Using LISTFILE

The LISTFILE command lists information about your CMS disk files. You can use this information within your EXEC.

```
/* Example using LISTFILE to find fileid of the first file    */
/* that matches a given filename.                             */
        address command
           'MAKEBUF'
           'LISTFILE' filename '* * (FIFO'
        if rc¬=0 then filetype='EXEC'
        else pull filename filetype filemode .
        address command 'DROPBUF'
```

Using NAMEFIND

The NAMEFIND command displays/stacks information from a NAMES file (default 'userid NAMES'). Following is an example of how you can use the CMS NAMEFIND command in an EXEC:

```
/* Program to retrieve phone numbers */
arg nick .
'NAMEFIND :NICK' nick ':PHONE :NAME (LIFO'
if rc¬=0 then do
   say 'Sorry, no phone listing for' nick
   exit
   end
parse pull name
parse pull phone
if phone='' then do
   say 'Sorry, no phone listing for' name
   exit
   end
say name"'s phone number is" phone'.'
exit
```

Using QUERY and RDR

The following example illustrates one way that you can use the information returned from the QUERY and RDR commands in an EXEC:

```
/*  Sample exec to show QUERY and RDR command uses                    */

/*  This section uses the CMS QUERY command to stack information on
    the contents of the user's A-disk.  Then, it reads in
    the information (throwing away the header line stacked by
    the QUERY command) and prints out a formatted message.  Unused
    variables set by the PULL command can be displayed if you desire */

query disk a '(' lifo                        /* get disk information */
pull label cuu m stat cyl type,              /* read from the stack, */
     blksize files used '-' percent,         /*   separate into all  */
     left total .                            /*   variables          */
pull .                                       /* read header line     */
used = strip(used,l)                         /* strip leading blanks */
say 'The A-disk is' percent'% full ('used' used blocks out of' total,
    'available)'

/*  This section invokes the CMS RDR command which sets a
    return code depending on the status of the reader and also on
    the type of file in the reader, should one exist.  The System
    Product interpreter sets the variable RC to this returned
    value.  Next, depending on the returned value, this exec
    selectively executes one of several commands.                    */

rdr '('notype                                /* get info on rdr file  */
                                 /* RC set to return code from RDR command */
select
  when rc=0 then say 'Reader is empty'
  when rc=22 then disk load
  when rc=13 then say 'Reader is not ready'
  otherwise
    say 'Return code other than expected'
end
```

Using SET EXECTRAC

You can trace your System Product Interpreter or EXEC 2 EXEC by specifying SET EXECTRAC ON prior to EXEC invocation. To turn tracing off, specify SET EXECTRAC OFF.

Using Windows and Virtual Screens

WAITREAD VSCREEN is an important command for EXECs that read from and write to windows. A typical sequence for such an EXEC would be:

- Define a window and virtual screen (DEFINE WINDOW and DEFINE VSCREEN commands)
- Connect the window to the virtual screen (SHOW WINDOW command)
- Write data to the virtual screen (WRITE VSCREEN command)

- Issue the WAITREAD VSCREEN command

When an attention interrupt is received, the EXEC can process the WAITREAD. *n* variables and update the virtual screen using the WRITE VSCREEN command.

For more information on these commands, refer to the *VM/SP CMS Command Reference.*

Using XEDIT

You can use the XEDIT command within an EXEC and stack XEDIT subcommands to manipulate a file.

Writing XEDIT macros

Writing an XEDIT macro is like creating a new XEDIT subcommand. An XEDIT macro is a System Product Interpreter or EXEC 2 file invoked from the XEDIT environment.

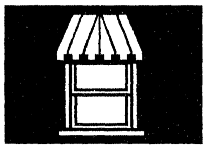
Refer to the *VM/SP System Product Editor User's Guide* for information on writing XEDIT macros using the System Product Interpreter. For information about XEDIT macros written in EXEC 2 language, refer to the *VM/SP EXEC 2 Reference.*

Exchanging Data Between Programs Through the Stack

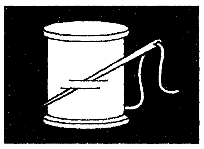
Refer to the *VM/SP System Product Interpreter User's Guide* for information on those REXX instructions which put data into the program stack.

Part 4: Tailoring Your System

VM/SP provides several ways for you to tailor your virtual machine. You can use full-screen CMS and windowing commands to customize your CMS session, or you can tailor the HELP facility to conform to your particular needs.



Chapter 15: Customizing Full-Screen CMS describes ways in which you can change the attributes of your windows and virtual screens.



Chapter 16: Tailoring the HELP Facility describes ways in which you can tailor the HELP Facility to your needs and describes techniques provided by the HELP Facility for creating user HELP description files.

Chapter 15. Customizing Full-Screen CMS

As you become more familiar with full-screen CMS, you may want to tailor windows and virtual screens to your special needs. First, it is necessary to understand what happens when you issue the command, SET FULLSCREEN ON.

During full-screen CMS initialization:

- All default virtual screens that you have not defined are defined, such as a virtual screen for CMS and CP output and virtual screens for messages, network messages, warnings from the operator, and status information.
- All default windows that you have not defined are defined, with the exception of the WM window. The WM window is defined when you specify the command POP WINDOW WM, when you press the PA1 key, or when the window is automatically displayed on your screen. (Refer to Chapter 9, "Looking at VM/SP Through Windows" for detailed information on when the WM window is displayed).
- All the reserved areas for the default virtual screens are written.
- Default windows are connected to the appropriate virtual screens.
- CMSPF key definitions are established.
- A connection to the IUCV Message All System Service is established and various message classes are routed to appropriate virtual screens.
- Logging is started for the MESSAGE and WARNING virtual screens. Messages are logged into the file MESSAGE LOGFILE, and warnings are logged into WARNING LOGFILE.
- The cursor is set in the CMS virtual screen.
- The CP TERMINAL BRKKEY NONE command is issued.

For more information, refer to the SET FULLSCREEN and SET CMSPF commands in the *VM/SP CMS Command Reference.*

You may want to execute certain commands before entering full-screen CMS. Other commands may be executed after setting full-screen CMS on. For example, if you want to change the size of a default virtual screen or change the definition of a default window, you must enter the appropriate commands before setting full-screen CMS on. The following chapter teaches you how to easily change a default setting.

You should read this section at the terminal so you can try the exercises as you read the text. If you want to find out more about a command, refer to the *VM/SP CMS Command Reference.*

Tailoring System Defaults

By defining windows and virtual screens before entering full-screen CMS, you can override full-screen CMS default definitions. (See the tables at the end of this chapter for default values). Once in full-screen CMS, you can change features such as virtual screen reserved lines, border characters, and the CMSPF keys.

Let's change the MESSAGE window and MESSAGE virtual screen to see how tailoring works. If you are in full-screen CMS, set full-screen off by entering:

```
set fullscreen off
```

Let's define the MESSAGE virtual screen to be 35 lines by 70 columns, which means it will be larger than the default of 20 lines by 70 columns. We will put 2 reserved lines at the top (for a title) and no reserved lines at the bottom. We will also specify the default options for the MESSAGE virtual screen: SYSTEM, PROTECT, and WHITE. Refer to the tables at the end of this chapter for a description of these options. Enter the command as follows:

```
define vscreen message 35 70 2 0 (system protect white
```

Next, let's define the MESSAGE window to be 10 lines by 71 columns, which means it will be larger than the default of 8 lines by 71 columns. It will be located at line 10 and column 5 on your physical screen. (The default location is line 11 and column 3). Again, we will specify the default options: SYSTEM, VARIABLE, and POP. Enter the command as follows:

```
define window message 10 71 10 5 (system variable pop
```

Now, let's change the border character on the MESSAGE window to a "$" for all sides of the window border. By default, the top and bottom characters are "-" (dash) and the right and left sides are "|" (vertical bar). To do this, enter:

```
set border message on (all $
```

Here's where you set full-screen CMS on. Enter:

```
set fullscreen on
```

Next, define a field at reserved line 1 and column 1 of the MESSAGE virtual screen with a length of 70 (the number of columns in the virtual screen). This prepares the MESSAGE virtual screen for a new title by replacing the default title with blanks. Thus, you don't have to worry about trying to overlay existing data.

Enter the command:

```
write vscreen message 1 1 70 (reserved blank
```

Finally, let's change the title of the MESSAGE virtual screen. With the WRITE VSCREEN command, we will change the title from the default title "MESSAGES" to "My Personal Messages." The new title will begin in column 26 and be 20 characters long.

```
write vscreen message 1 26 20 (reserved data My Personal Messages
```

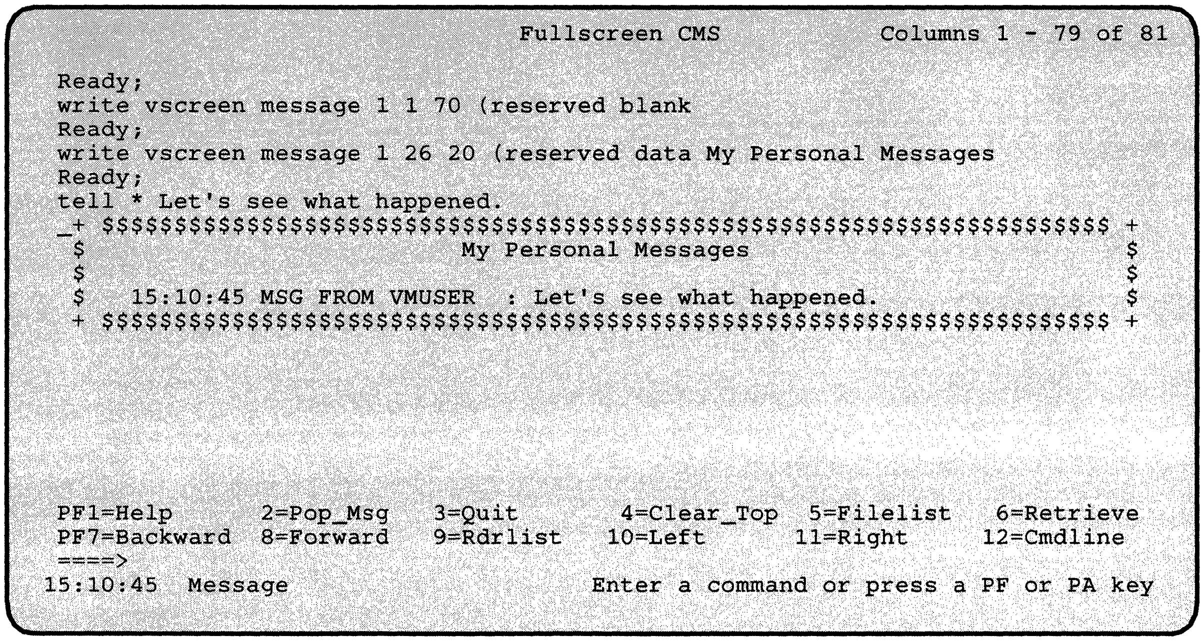
Now you have tailored the MESSAGE window and virtual screen to your own specifications. Another way to accomplish this would be to write an EXEC. Here's what the EXEC would look like:

```
/* EXEC to tailor the MESSAGE virtual screen and window */
'define vscreen message 35 70 2 0 (system protect white'
'define window message 10 71 10 5 (system variable pop'
'set border message on (all $'
'set fullscreen on'
'write vscreen message 1 1 70 (reserved blank'
'write vscreen message 1 26 20 (reserved data My Personal Messages'
```

To see your new MESSAGE window, send this message to yourself:

```
tell * Let's see what happened.
```

When the MESSAGE window is popped, you see the following:



```
                                  Fullscreen CMS          Columns 1 - 79 of 81
  Ready;
  write vscreen message 1 1 70 (reserved blank
  Ready;
  write vscreen message 1 26 20 (reserved data My Personal Messages
  Ready;
  tell * Let's see what happened.
  _+ $$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$ +
   $                          My Personal Messages                              $
   $                                                                            $
   $   15:10:45 MSG FROM VMUSER  : Let's see what happened.                     $
   + $$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$ +









  PF1=Help      2=Pop_Msg   3=Quit      4=Clear_Top  5=Filelist   6=Retrieve
  PF7=Backward  8=Forward   9=Rdrlist  10=Left      11=Right     12=Cmdline
  ====>
 15:10:45  Message                    Enter a command or press a PF or PA key
```

Figure 56. Your Newly-Defined MESSAGE Window

Remember, because you defined the window as variable in size, the size of the window will vary depending on how many messages you receive.

Now, enter a "D" in any corner of the window to drop it from your screen.

You may wish to tailor your CMS session by writing other simple EXECs to perform windowing functions. For example, you may find it useful to set a PF key to toggle between popping and dropping the MESSAGE window. To try this example, create a file with a filename of POPDROP and filetype of EXEC. Then, XEDIT the file and enter the following:

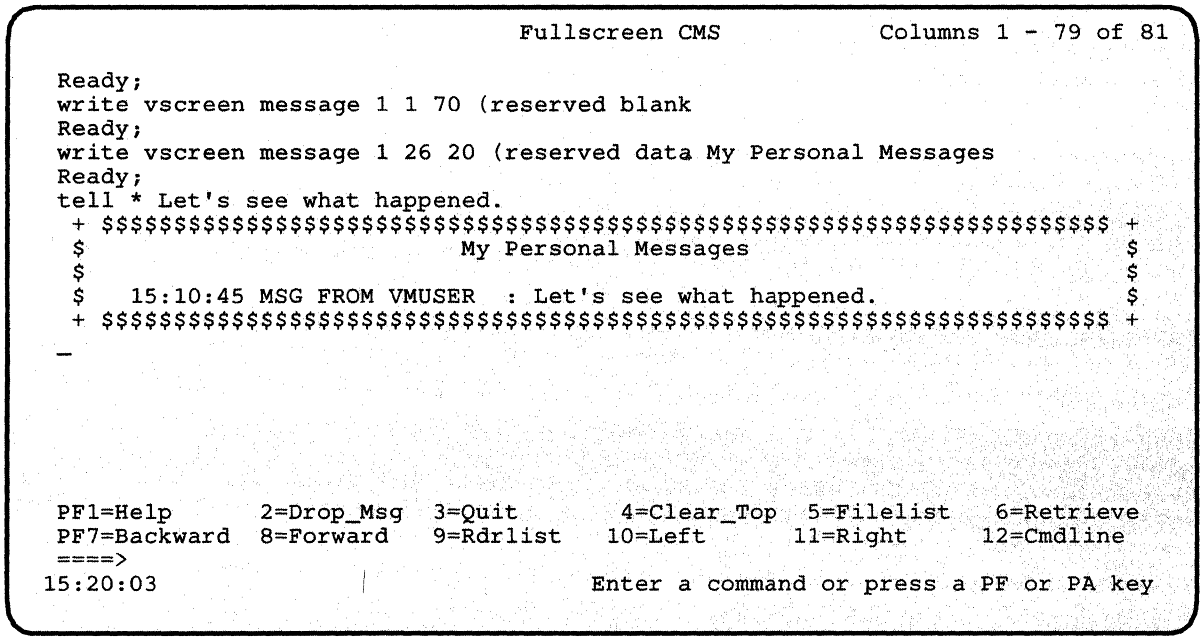
```
/* EXEC to set a PF key to POP and DROP the MESSAGE window*/
 PARSE UPPER ARG whichway
 ADDRESS COMMAND
 IF whichway ¬= "DROP" THEN
   DO
     'POP WINDOW MESSAGE'
     'SET CMSPF 02 Drop_Msg NOECHO POPDROP DROP'
   END
ELSE
   DO
     'DROP WINDOW MESSAGE'
     'SET CMSPF 02 Pop_Msg NOECHO POPDROP POP'
   END
```

File the EXEC. Now, when you issue the command POPDROP, CMSPF 2 will initially be set to drop the MESSAGE window. Therefore, when you receive a message, and the MESSAGE window pops, you can simply press CMSPF 2 to drop it.

Let's try it! Type:

```
popdrop
```

to issue the POPDROP EXEC. The EXEC will pop the MESSAGE window. The window will appear again as shown below:



```
                                   Fullscreen CMS         Columns 1 - 79 of 81

  Ready;
  write vscreen message 1 1 70 (reserved blank
  Ready;
  write vscreen message 1 26 20 (reserved data My Personal Messages
  Ready;
  tell * Let's see what happened.
   + $$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$ +
   $                        My Personal Messages                             $
   $                                                                         $
   $   15:10:45 MSG FROM VMUSER  : Let's see what happened.                  $
   + $$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$ +
  _






  PF1=Help      2=Drop_Msg  3=Quit      4=Clear_Top  5=Filelist   6=Retrieve
  PF7=Backward  8=Forward   9=Rdrlist  10=Left      11=Right     12=Cmdline
  ====>
 15:20:03                              Enter a command or press a PF or PA key
```

Figure 57. Using the POPDROP EXEC

Press CMSPF 2 to drop the window. From now on, CMSPF 2 will toggle between popping and dropping the MESSAGE window. When the MESSAGE window is showing on your screen, you can press CMSPF 2 to drop it. When the MESSAGE window is not showing on your screen, you can press CMSPF 2 to pop the window.

Note: In order for the MESSAGE window to appear on your screen, it must contain at least one message. If you try to pop the MESSAGE window before any messages have been received, the window will not be displayed.

Press CMSPF 2 to pop the MESSAGE window and prepare for the next exercise.

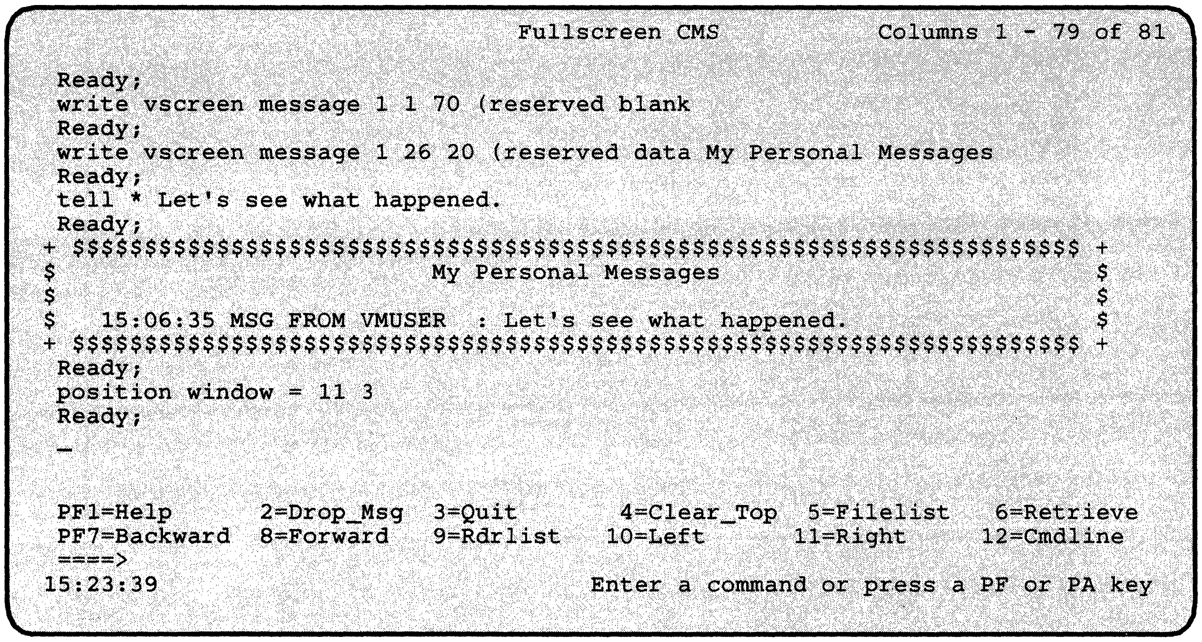
POSITION WINDOW

With the POSITION WINDOW command, you can move a window anywhere on the physical screen. Let's move our new MESSAGE window.

If you use an "=" in the command syntax instead of the window name, the command moves the topmost window. Since the MESSAGE window is currently the topmost window, you can enter the following command to move it:

```
position window = 11 3
```

The 11 and 3 are the line and column, respectively, where the window's UPPER left corner will be repositioned relative to the TOP of the screen. When you enter this command, the MESSAGE window is repositioned on the screen as shown:



```
                              Fullscreen CMS           Columns 1 - 79 of 81

 Ready;
 write vscreen message 1 1 70 (reserved blank
 Ready;
 write vscreen message 1 26 20 (reserved data My Personal Messages
 Ready;
 tell * Let's see what happened.
 Ready;
+ $$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$ +
$                          My Personal Messages                          $
$                                                                        $
$   15:06:35 MSG FROM VMUSER  : Let's see what happened.                 $
+ $$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$ +
 Ready;
 position window = 11 3
 Ready;
 _


 PF1=Help      2=Drop_Msg  3=Quit      4=Clear_Top  5=Filelist   6=Retrieve
 PF7=Backward  8=Forward   9=Rdrlist  10=Left      11=Right     12=Cmdline
 ====>
15:23:39                              Enter a command or press a PF or PA key
```

Figure 58. Using the POSITION WINDOW Command

SIZE WINDOW

If you wish, you can change the maximum number of lines in your MESSAGE window with the SIZE WINDOW command. To do this, enter:

```
size window = 8 70
```

Since the MESSAGE window is variable in size, you may not see the result of the change in size until you receive more messages which fill and expand the window.

MAXIMIZE WINDOW and RESTORE WINDOW

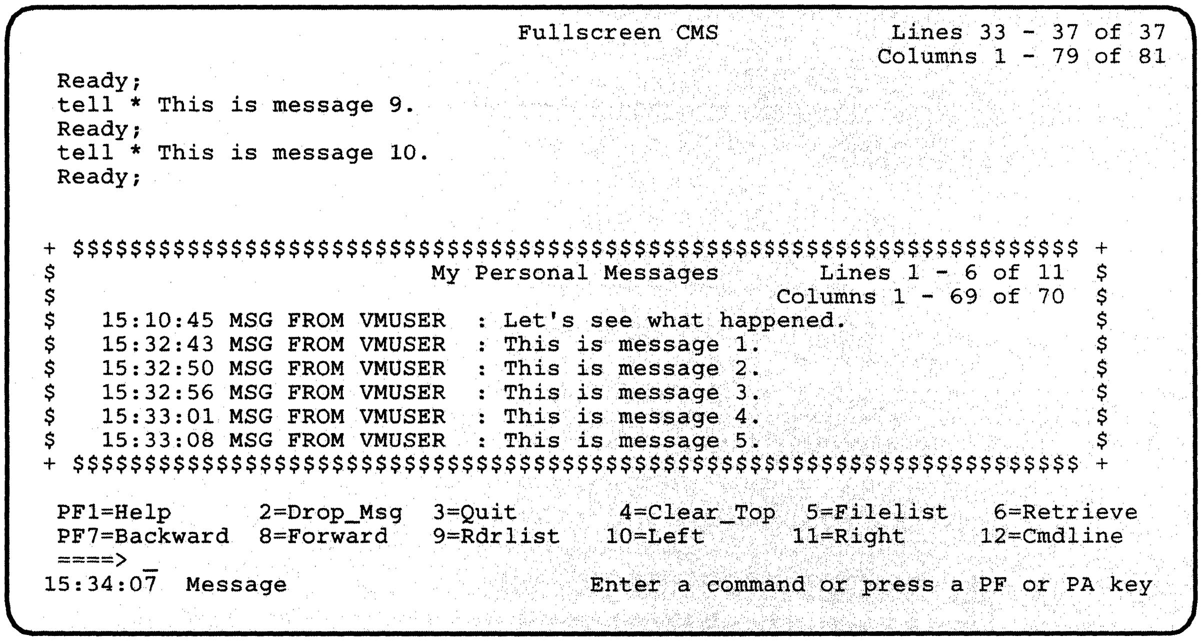
Suppose you wanted enlarge a window to look at more of the data contained in the virtual screen. With the MAXIMIZE WINDOW command, you can expand the size of the window.

If you are following the exercises, the MESSAGE window is still on your screen. It has the title: "My Personal Messages." It also has one message in it: "Let's see what happened."

To see what happens with the MAXIMIZE WINDOW command, send yourself the following 10 messages (press ENTER after each message):

```
tell * This is message 1.
tell * This is message 2.
tell * This is message 3.
tell * This is message 4.
tell * This is message 5.
tell * This is message 6.
tell * This is message 7.
tell * This is message 8.
tell * This is message 9.
tell * This is message 10.
```

After you scroll forward, your screen will appear as shown below:



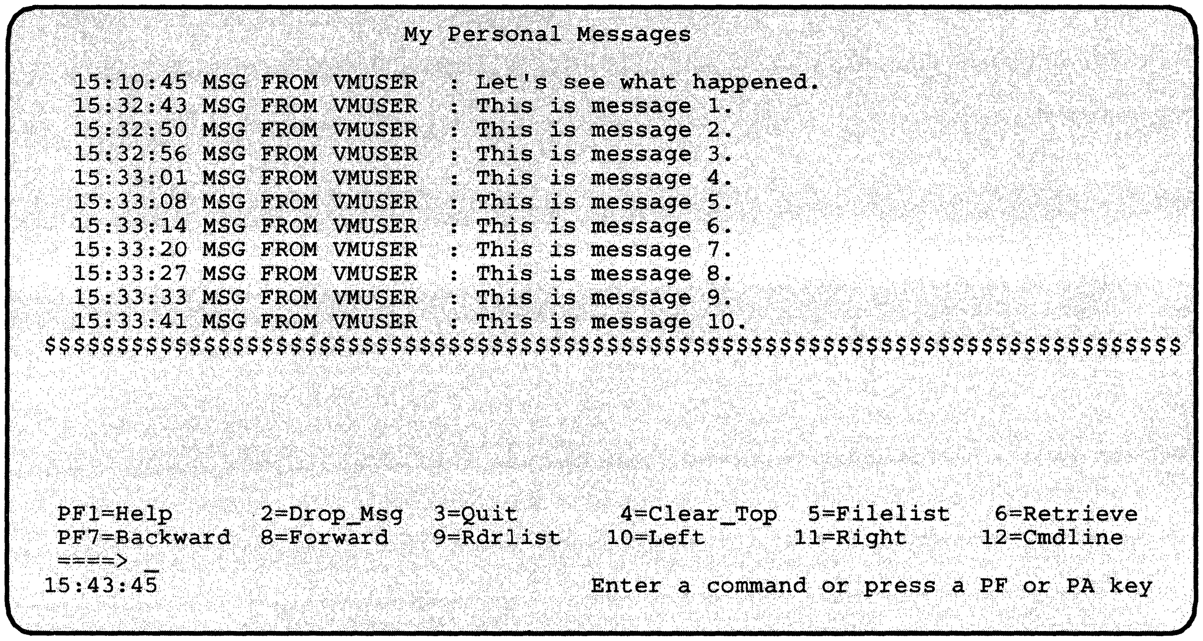
```
                              Fullscreen CMS          Lines 33 - 37 of 37
                                                      Columns 1 - 79 of 81
 Ready;
 tell * This is message 9.
 Ready;
 tell * This is message 10.
 Ready;


+ $$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$ +
$                          My Personal Messages        Lines 1 - 6 of 11   $
$                                                   Columns 1 - 69 of 70   $
$   15:10:45 MSG FROM VMUSER  : Let's see what happened.                   $
$   15:32:43 MSG FROM VMUSER  : This is message 1.                         $
$   15:32:50 MSG FROM VMUSER  : This is message 2.                         $
$   15:32:56 MSG FROM VMUSER  : This is message 3.                         $
$   15:33:01 MSG FROM VMUSER  : This is message 4.                         $
$   15:33:08 MSG FROM VMUSER  : This is message 5.                         $
+ $$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$ +

 PF1=Help      2=Drop_Msg  3=Quit       4=Clear_Top  5=Filelist   6=Retrieve
 PF7=Backward  8=Forward   9=Rdrlist   10=Left      11=Right     12=Cmdline
====> _
15:34:07  Message                      Enter a command or press a PF or PA key
```

Figure 59. "Popping" the MESSAGE Window

Notice that the location indicator in the upper right corner of the window shows "Lines 1 - 6 of 11." To see the remaining messages, let's maximize the MESSAGE window. Enter:

```
maximize window message
```

the MESSAGE window now looks like this:



```
                          My Personal Messages

   15:10:45 MSG FROM VMUSER  : Let's see what happened.
   15:32:43 MSG FROM VMUSER  : This is message 1.
   15:32:50 MSG FROM VMUSER  : This is message 2.
   15:32:56 MSG FROM VMUSER  : This is message 3.
   15:33:01 MSG FROM VMUSER  : This is message 4.
   15:33:08 MSG FROM VMUSER  : This is message 5.
   15:33:14 MSG FROM VMUSER  : This is message 6.
   15:33:20 MSG FROM VMUSER  : This is message 7.
   15:33:27 MSG FROM VMUSER  : This is message 8.
   15:33:33 MSG FROM VMUSER  : This is message 9.
   15:33:41 MSG FROM VMUSER  : This is message 10.
 $$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$




  PF1=Help      2=Drop_Msg  3=Quit       4=Clear_Top  5=Filelist   6=Retrieve
  PF7=Backward  8=Forward   9=Rdrlist   10=Left      11=Right     12=Cmdline
  ====> _
 15:43:45                               Enter a command or press a PF or PA key
```

Figure 60. "Maximizing" a Window

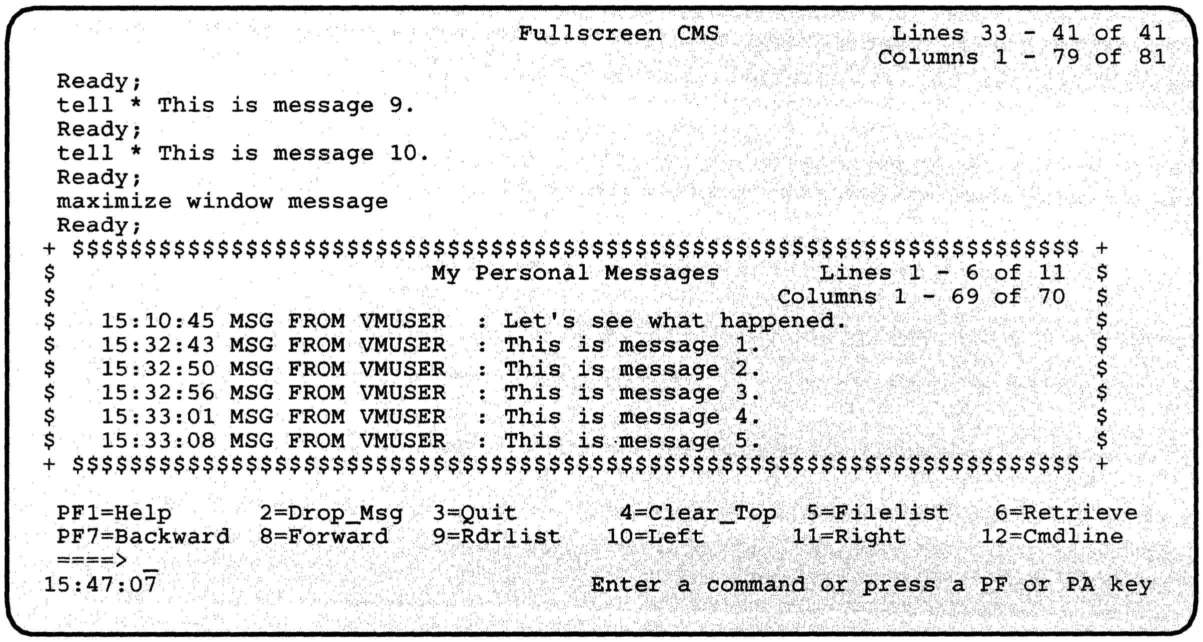
Even though the window is maximized, it doesn't fill the entire screen because the MESSAGE window is variable in size. It expands depending on how much data there is to display. You will notice that the window moved to the location of line 1 and column 1 on the screen and you can see all the messages.

If you maximized a window so that the window filled the entire screen, you might not be able to enter commands if the window is protected. In this instance, the WM window would be automatically displayed, and the WMPF keys and command line would be available to manipulate the window. For more information on the WM window, refer to Chapter 9, "Looking at VM/SP Through Windows."

To return the window to its previous size and location on the screen, you can enter the RESTORE WINDOW command:

```
restore window message
```

Here's how your screen looks (the same as it was prior to the MAXIMIZE WINDOW):



```
                              Fullscreen CMS             Lines 33 - 41 of 41
                                                       Columns 1 - 79 of 81
 Ready;
 tell * This is message 9.
 Ready;
 tell * This is message 10.
 Ready;
 maximize window message
 Ready;
+ $$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$ +
$                         My Personal Messages       Lines 1 - 6 of 11  $
$                                                 Columns 1 - 69 of 70  $
$   15:10:45 MSG FROM VMUSER  : Let's see what happened.                $
$   15:32:43 MSG FROM VMUSER  : This is message 1.                      $
$   15:32:50 MSG FROM VMUSER  : This is message 2.                      $
$   15:32:56 MSG FROM VMUSER  : This is message 3.                      $
$   15:33:01 MSG FROM VMUSER  : This is message 4.                      $
$   15:33:08 MSG FROM VMUSER  : This is message 5.                      $
+ $$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$ +

 PF1=Help      2=Drop_Msg  3=Quit      4=Clear_Top  5=Filelist   6=Retrieve
 PF7=Backward  8=Forward   9=Rdrlist   10=Left      11=Right     12=Cmdline
 ====> _
15:47:07                               Enter a command or press a PF or PA key
```

Figure 61. The Window after RESTORE WINDOW

Using the SET Command

The following sections provide information and examples on using SET commands.

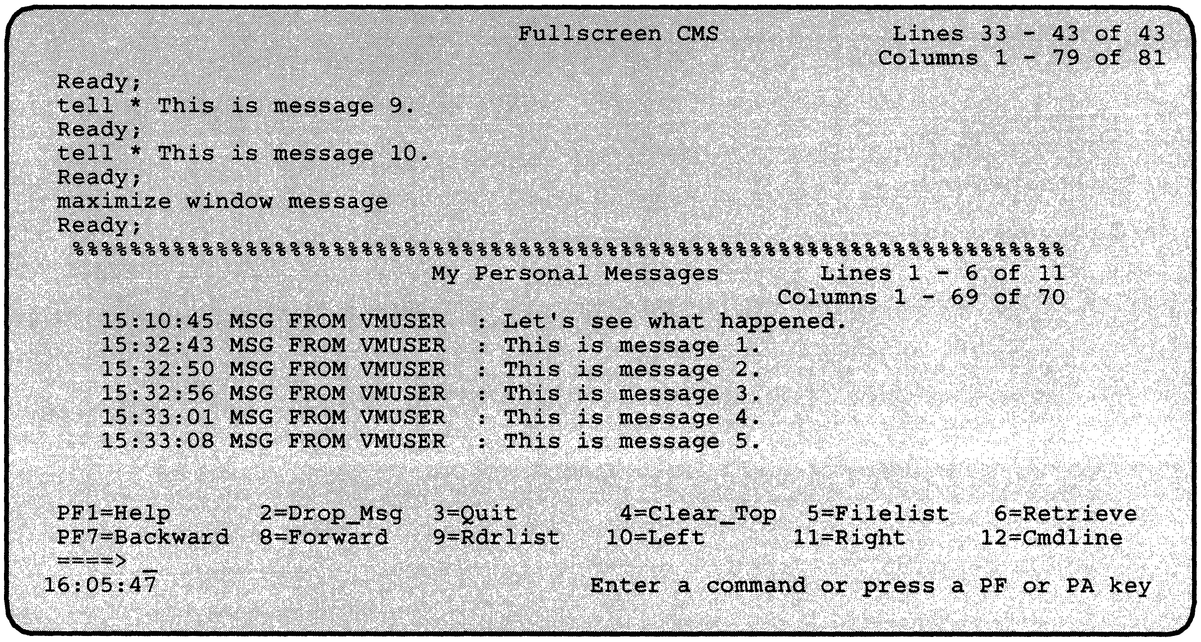
SET BORDER

With the SET BORDER command, you can tailor the characters of a border or change how a border is displayed. These features help you to visually separate information displayed in different windows. With the earlier example, you changed all the borders to $. Now let's change just the top border to %.

When you issue the command for only the top border, the top edge changes to %, but you will not see the other sides of the border. For example, enter this command:

```
set border message on (top %
```

The border looks like this:



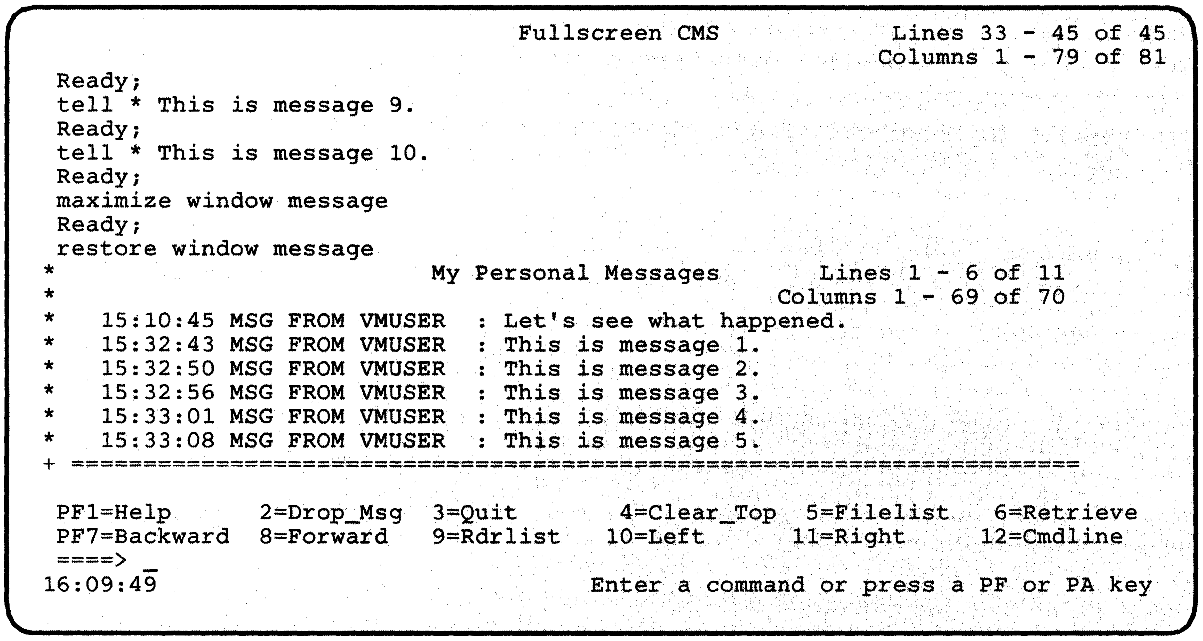
```
                                 Fullscreen CMS          Lines 33 - 43 of 43
                                                       Columns 1 - 79 of 81
  Ready;
  tell * This is message 9.
  Ready;
  tell * This is message 10.
  Ready;
  maximize window message
  Ready;
   %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
                           My Personal Messages       Lines 1 - 6 of 11
                                                    Columns 1 - 69 of 70
     15:10:45 MSG FROM VMUSER  : Let's see what happened.
     15:32:43 MSG FROM VMUSER  : This is message 1.
     15:32:50 MSG FROM VMUSER  : This is message 2.
     15:32:56 MSG FROM VMUSER  : This is message 3.
     15:33:01 MSG FROM VMUSER  : This is message 4.
     15:33:08 MSG FROM VMUSER  : This is message 5.


  PF1=Help      2=Drop_Msg  3=Quit       4=Clear_Top  5=Filelist   6=Retrieve
  PF7=Backward  8=Forward   9=Rdrlist   10=Left      11=Right     12=Cmdline
  ====> _
 16:05:47                                Enter a command or press a PF or PA key
```

Figure 62. Changing Only the Top Border

To see just the bottom and left borders, enter:

```
set border message on (bottom = left *
```

Here's what happens to the window:



```
                                 Fullscreen CMS              Lines 33 - 45 of 45
                                                            Columns 1 - 79 of 81
 Ready;
 tell * This is message 9.
 Ready;
 tell * This is message 10.
 Ready;
 maximize window message
 Ready;
 restore window message
*                          My Personal Messages      Lines 1 - 6 of 11
*                                                  Columns 1 - 69 of 70
*   15:10:45 MSG FROM VMUSER  : Let's see what happened.
*   15:32:43 MSG FROM VMUSER  : This is message 1.
*   15:32:50 MSG FROM VMUSER  : This is message 2.
*   15:32:56 MSG FROM VMUSER  : This is message 3.
*   15:33:01 MSG FROM VMUSER  : This is message 4.
*   15:33:08 MSG FROM VMUSER  : This is message 5.
+ ====================================================================

 PF1=Help      2=Drop_Msg  3=Quit       4=Clear_Top  5=Filelist   6=Retrieve
 PF7=Backward  8=Forward   9=Rdrlist   10=Left      11=Right     12=Cmdline
 ====> _
16:09:49                                 Enter a command or press a PF or PA key
```

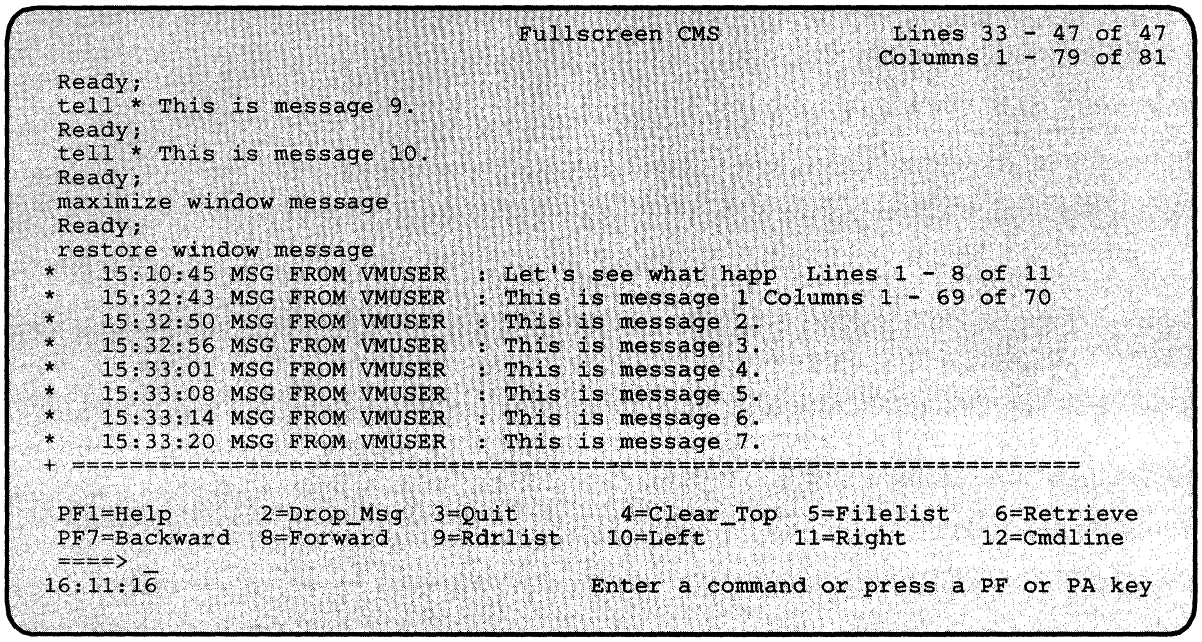
Figure 63. Changing the Bottom and Left Window Borders

SET RESERVED

Suppose you don't want to see a title in a window. With the SET RESERVED command, you can delete the title "My Personal Messages" from the MESSAGE window and re-use the area previously reserved for the title. Enter:

```
set reserved message 0 0
```

Here's what happens:



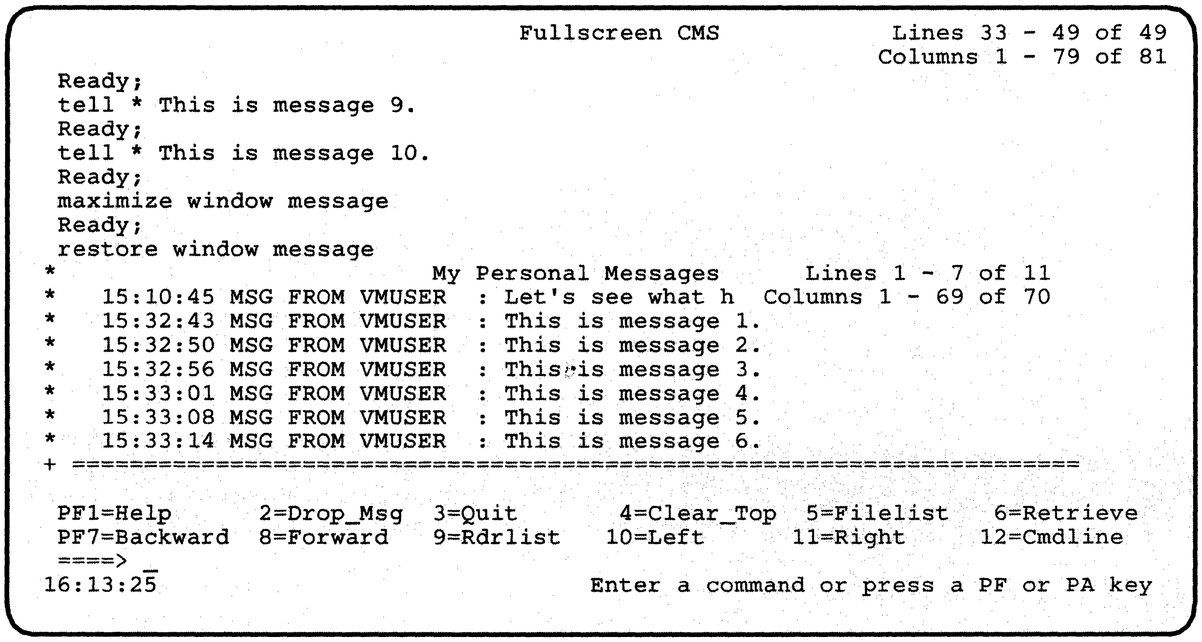
```
                             Fullscreen CMS          Lines 33 - 47 of 47
                                                   Columns 1 - 79 of 81
 Ready;
 tell * This is message 9.
 Ready;
 tell * This is message 10.
 Ready;
 maximize window message
 Ready;
 restore window message
*   15:10:45 MSG FROM VMUSER  : Let's see what happ  Lines 1 - 8 of 11
*   15:32:43 MSG FROM VMUSER  : This is message 1 Columns 1 - 69 of 70
*   15:32:50 MSG FROM VMUSER  : This is message 2.
*   15:32:56 MSG FROM VMUSER  : This is message 3.
*   15:33:01 MSG FROM VMUSER  : This is message 4.
*   15:33:08 MSG FROM VMUSER  : This is message 5.
*   15:33:14 MSG FROM VMUSER  : This is message 6.
*   15:33:20 MSG FROM VMUSER  : This is message 7.
+ ======================================================================

 PF1=Help      2=Drop_Msg  3=Quit      4=Clear_Top  5=Filelist   6=Retrieve
 PF7=Backward  8=Forward   9=Rdrlist  10=Left      11=Right     12=Cmdline
 ====> _
16:11:16                                Enter a command or press a PF or PA key
```

Figure 64. Deleting a Window Title

Now, let's set the reserved line to 1 so there will not be a blank line after the window title. Enter:

```
set reserved message 1 0
```

Your screen will now look like this:



```
                               Fullscreen CMS           Lines 33 - 49 of 49
                                                        Columns 1 - 79 of 81
 Ready;
 tell * This is message 9.
 Ready;
 tell * This is message 10.
 Ready;
 maximize window message
 Ready;
 restore window message
*                          My Personal Messages     Lines 1 - 7 of 11
*   15:10:45 MSG FROM VMUSER  : Let's see what h  Columns 1 - 69 of 70
*   15:32:43 MSG FROM VMUSER  : This is message 1.
*   15:32:50 MSG FROM VMUSER  : This is message 2.
*   15:32:56 MSG FROM VMUSER  : This is message 3.
*   15:33:01 MSG FROM VMUSER  : This is message 4.
*   15:33:08 MSG FROM VMUSER  : This is message 5.
*   15:33:14 MSG FROM VMUSER  : This is message 6.
+ ======================================================================

 PF1=Help      2=Drop_Msg  3=Quit      4=Clear_Top  5=Filelist   6=Retrieve
 PF7=Backward  8=Forward   9=Rdrlist  10=Left      11=Right     12=Cmdline
 ====> _
16:13:25                          Enter a command or press a PF or PA key
```

Figure 65. Deleting a Blank Reserved Line

Now, drop the MESSAGE by typing a "D" in any corner. Then, enter the command:

```
clear vscreen message
```

to clear the MESSAGE virtual screen of all old messages.

SET WINDOW

By using the SET WINDOW command, you can choose whether or not you want the MESSAGE window to pop when you receive a message. The default setting is for the window to pop. In addition, the message class indicator will be updated, and the terminal alarm will sound. You can change this by setting the window to NOPOP.

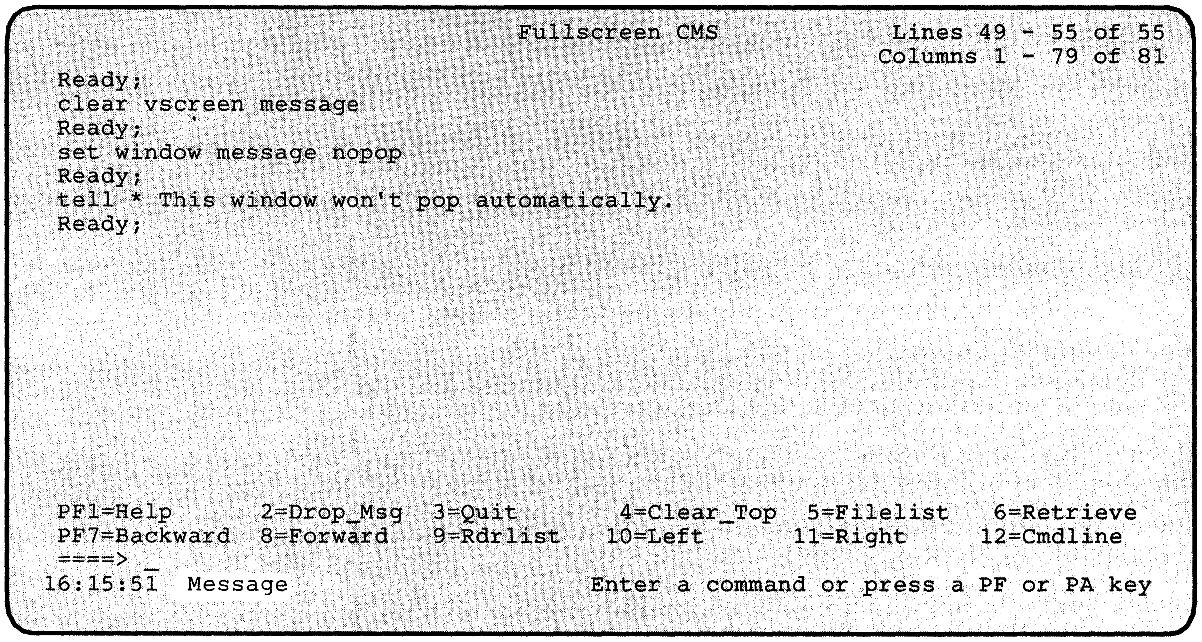
To try it, enter:

```
set window message nopop
```

Now send yourself the following message:

```
tell * This window won't pop automatically.
```

After you press ENTER, you will notice that your terminal alarm will sound and the message class indicator will be updated as shown in the example below. However, the window will not be automatically displayed on your screen. At this point, you would need to enter the command POP WINDOW MESSAGE in order to pop the window.



```
                               Fullscreen CMS          Lines 49 - 55 of 55
                                                       Columns 1 - 79 of 81
  Ready;
  clear vscreen message
  Ready;
  set window message nopop
  Ready;
  tell * This window won't pop automatically.
  Ready;








  PF1=Help      2=Drop_Msg  3=Quit       4=Clear_Top  5=Filelist   6=Retrieve
  PF7=Backward  8=Forward   9=Rdrlist   10=Left      11=Right     12=Cmdline
  ====> _
 16:15:51  Message                     Enter a command or press a PF or PA key
```

Figure 66. Message Class Indicator

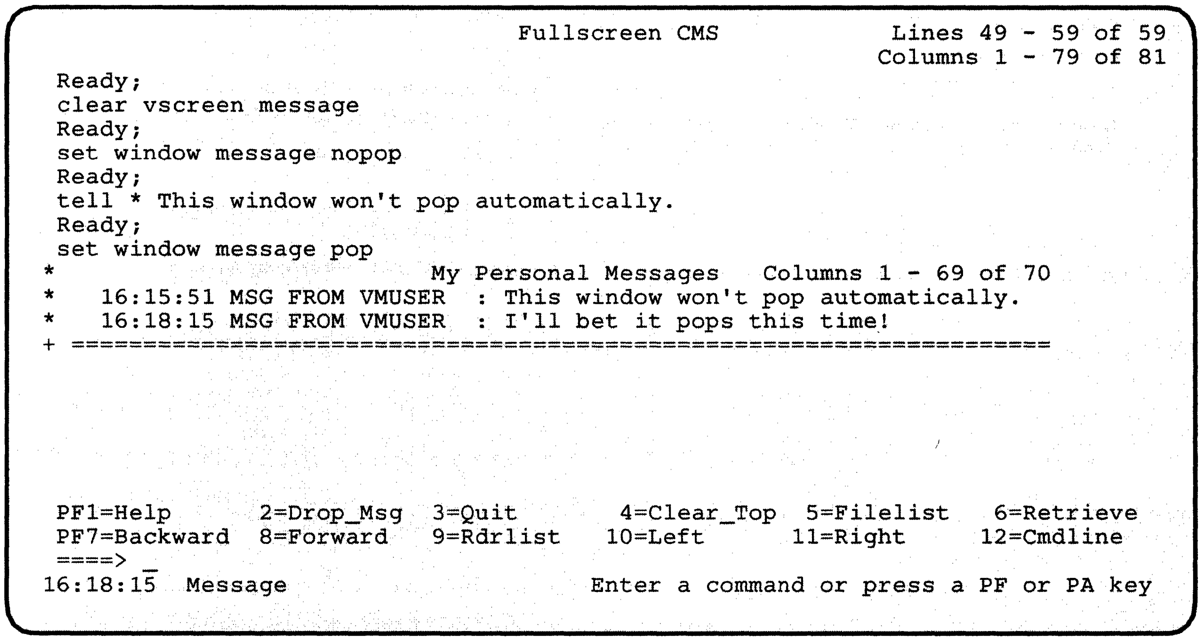
Now let's set the window to pop when you receive a message. Enter:

```
set window message pop
```

Send yourself the following message:

```
tell * I'll bet it pops this time!
```

Your screen will display the MESSAGE window as shown below:



```
                               Fullscreen CMS          Lines 49 - 59 of 59
                                                       Columns 1 - 79 of 81
 Ready;
 clear vscreen message
 Ready;
 set window message nopop
 Ready;
 tell * This window won't pop automatically.
 Ready;
 set window message pop
*                        My Personal Messages   Columns 1 - 69 of 70
*   16:15:51 MSG FROM VMUSER  : This window won't pop automatically.
*   16:18:15 MSG FROM VMUSER  : I'll bet it pops this time!
+ ====================================================================





 PF1=Help      2=Drop_Msg  3=Quit       4=Clear_Top  5=Filelist   6=Retrieve
 PF7=Backward  8=Forward   9=Rdrlist   10=Left      11=Right     12=Cmdline
 ====> _
16:18:15  Message                     Enter a command or press a PF or PA key
```

Figure 67. The MESSAGE Window

Now, clear the MESSAGE window with the "C" border command, or press PA2 to scroll and drop the MESSAGE window. This brings you back to the CMS window. If you want to leave full-screen CMS, enter the command: SET FULLSCREEN OFF. You can also enter SET FULLSCREEN SUSPEND or press CMSPF 3 to suspend full-screen CMS. If you suspend full-screen CMS, you can later resume your CMS session where you left off. None of the default or user-defined settings for windows, virtual screens, or PF keys will be lost.

Exploring Other Commands

These are just a few ways you can tailor your windows and virtual screens. By looking at the *VM/SP CMS Command Reference* and trying some commands, you can make windows and virtual screens work for you.

For further information which may be useful when you begin using windowing and full-screen CMS, refer to Appendix D, "Considerations for Full-Screen CMS and Windowing."

Window and Virtual Screen Tables

If you are interested in tailoring windows or virtual screens, you may find it useful to refer to the following tables for the full-screen CMS default characteristics for windows and virtual screens.

Window	Lines	Cols	Psline	Pscol	Options
STATUS	1	Pscr	-1	1	FIXED NOBORDER NOPOP NOTOP
CMS	Pscr	Pscr	1	1	FIXED BORDER NOPOP TOP
NETWORK	8 (max.)	71	-12	7	VARIABLE BORDER NOPOP TOP
WARNING	6 (max.)	71	3	3	VARIABLE BORDER POP TOP
MESSAGE	8 (max.)	71	11	3	VARIABLE BORDER POP TOP
WM	5	Pscr	-1	1	FIXED BORDER NOPOP NOTOP
CMSOUT	8	75	9	3	VARIABLE BORDER POP TOP

Figure 68. Default Windows

NOTES:

Pscr
Size of the physical screen.

Psline
The line on the physical screen where the upper (when psline is positive) or lower (when psline is negative) corner of the window will be placed.

Pscol
The column on the physical screen where the upper left corner of the window will be placed.

FIXED
The number of lines in the window is always constant.

VARIABLE
The number of lines in the window may vary from zero to the maximum, depending on the amount of scrollable data to be displayed.

BORDER
The window borders are displayed when possible.

Note: In the case of the CMS window, even though the borders are on, you cannot see them because the window is the size of the physical screen.

NOBORDER
The window borders are not displayed.

POP
The window is displayed on top of all other windows when the virtual screen that the window is showing is updated.

NOPOP
There is no effect on the window's position in the ordered list of windows when the virtual screen that the window is showing is updated.

TOP
The window may qualify as the topmost window.

NOTOP
The window cannot qualify as the topmost window.

Although the WM window is a default window, it is not defined when you enter full-screen CMS. The WM window is defined when you issue the command POP WINDOW WM, when you press the PA1 key, or when the window is automatically displayed on your screen.

All default windows are SYSTEM windows; they are not lost when the system abnormally terminates (abends) or when the HX (halt execution) command is issued.

Virtual Screen	Lines	Cols	Rtop	Rbot	Dcolor	Options
WM	1	Pscr	0	5	White	NOPROTECT
STATUS	1	Pscr	0	0	White	PROTECT
NETWORK	16	70	2	0	Blue	PROTECT
WARNING	4	70	2	0	Red	PROTECT
MESSAGE	20	70	2	0	White	PROTECT
CMS	120	Pscr	2	5	Green	NOPROTECT

Figure 69. Default Virtual Screens

NOTES:

Pscr
Physical screen size. For terminals with a screen size of 80 columns or less, the CMS virtual screen contains 81 columns. For terminals with a screen size greater than 80 columns, the CMS virtual screen contains the same number of columns as the physical screen. The status and WM virtual screens always contain the same number of columns as the physical screen.

Rtop
Top reserved lines

Rbot
Bottom reserved lines

Dcolor
Data color (if your terminal is equipped for color)

PROTECT
You cannot type into the window(s) connected to the virtual screen because the data is protected.

NOPROTECT
You can type into the window(s) connected to virtual screen; the data is not protected.

Although the WM virtual screen is a default virtual screen, it is not defined when you enter full-screen CMS. The WM virtual screen is defined when you issue the command POP WINDOW WM, when you press the PA1 key, or when the window is automatically displayed on your screen.

All default virtual screens are TYPE virtual screens. TYPE means data is moved to the virtual screen when the virtual screen is updated. In addition, all default virtual screens are SYSTEM virtual screens. SYSTEM means the virtual screen is retained when the system

abnormally terminates (abends) or when the HX (halt execution) command is issued.

Considerations When Disconnecting and Reconnecting

If you disconnect from your terminal and later reconnect at a terminal which has a different physical screen size, you may notice certain changes.

Note: You must reconnect to a terminal with a physical screen size of at least 24 lines by 80 columns. If you reconnect to a terminal with a physical screen smaller than 24 by 80, full-screen CMS will be suspended. You will need to reconnect to a terminal with a larger physical screen to continue your full-screen CMS session.

Once you reconnect and resume your full-screen CMS session, certain windows and virtual screens which are the size of the physical screen will be resized, relocated, or redefined to fit the dimensions of the new physical screen.

The windows and virtual screens which may be affected are those listed in the previous tables with "Pscr" under the headings 'Lines' or 'Columns'.

The following list provides details on how these windows and virtual screens will change:

- Default windows are resized to fit the new physical screen.
- Default windows you have moved are resized and relocated to the default size and location.
- User-defined windows with a size and/or location that cannot fit on the new physical screen will be adjusted.
- Default virtual screens are redefined to fit the size of the new physical screen only if the width of the physical screen changes. Default reserved line data will be rewritten in the new virtual screens. In the instance where the virtual screens must be redefined, you will lose the data contained in those virtual screens.
- User-defined virtual screens will be untouched.

You may also notice after you reconnnect at another terminal that certain commands which depend upon the physical device characteristics will temporarily reflect the characteristics of the previous terminal. For example, such commands as QUERY DISPLAY and QUERY WINDOW (in the case where the window is the size of the physical screen) will not automatically reflect the physical screen size of the new terminal.

Once you set full-screen on or enter XEDIT (or a productivity aid which uses XEDIT), the settings relating to terminal size will be adjusted. Any subsequent commands you issue will reflect the physical screen size of your current terminal.

Message Routing Tables

You may also find it useful to refer to the following tables which contain information regarding the default settings for message routing.

Message Class	Virtual Screen	Options
CMS	CMS	NOALARM NONOTIFY
CP	CMS	NOALARM NONOTIFY
MESSAGE	MESSAGE	ALARM NOTIFY
WARNING	WARNING	ALARM NOTIFY
SCIF	MESSAGE	NOALARM NONOTIFY
NETWORK	NETWORK	NOALARM NOTIFY

Figure 70. Default Settings for Message Routing

ALARM
The alarm will sound when a message is received.

NOALARM
The alarm will not sound when a message is received.

NOTIFY
The message class indicator will be shown in the status area when you receive a message.

NONOTIFY
The virtual screen name will not be displayed in the status area when you receive a message.

Chapter 16. Tailoring the HELP Facility

One of the most useful features of the HELP facility is its flexibility. You can tailor the HELP displayed to suit your needs by simply creating or changing CMS files.

You might wish to generate new HELP files for any new commands, EXECs, or error messages you create. You may also wish to make updates to the VM/SP HELP files to fulfill your individual needs.

If you have your own set of HELP files, you can do as you wish with them. However, if you share a set of HELP files with other system users, you will have to get authority from the system administrator to alter the HELP facility.

Creating HELP Files

Each existing HELP file has a distinct filename and filetype. When you wish to write your own HELP files, there are naming conventions you must follow to ensure that your files will be recognized by the VM/SP HELP facility.

Filenames for HELP Files

When you change the existing HELP files or write your own HELP files, you must follow these filename conventions:

- The filename for components, commands, subcommands, or EXECs must be the exact full name of the component, command, subcommand, or EXEC.

- The filename for messages has the form xxxnnn(n)t, where:

xxx	is the component code prefix (for example, DMS for CMS messages). See *VM/SP System Messages and Codes* for a list of the component code prefixes.
nnn(n)	is the message number.
t	is the message type code (for example, E for error messages in CMS).

For example, the HELP filename for the CMS message

```
No filename specified
```

would be DMS001E.

- For commands or statement names containing special characters, HELP creates the filename by translating that special character.

  If the name is a single special character, then the filename will be the name of the special character. For example, "&" and "?" have the filenames of AMPRSAND and QUESMARK respectively. To display the HELP file XEDIT subcommand "?," you would issue HELP XEDIT ?. However, the actual fileid for that file would be QUESMARK HELPXEDI. The special characters are translated as follows:

CHARACTER	TRANSLATED TO
?	QUESMARK
=	EQUAL
/	SLASH
"	DBLQUOTE
&	AMPRSAND
*	ASTERISK
.	PERIOD
<	LESSTHAN
>	MORETHAN

  In command names or filenames with more than one character, replace the special character with the first letter of its name. For example, "&" (AMPRSAND) would be replaced by an "A." An exception is "*," which is replaced by an "R."

  Thus, the EXEC 2 statements &STACK and &EXIT would have the filenames ASTACK and AEXIT. Remember that these changes only apply to the filenames of the statements; they do not affect the way you call for a HELP file display. To display the HELP files for &STACK and &EXIT, you would issue HELP EXEC2 &STACK and HELP EXEC2 &EXIT.

Filetypes for HELP Files

You must also follow certain conventions for filetypes when you create your own HELP files:

- The filetype of the HELP file is HELPxxxx, where 'xxxx' is the name of the component to which the file belongs. For example, the filetype for a CMS command would be HELPCMS.

- The filetype for subcommands is HELPxxxx, where 'xxxx' identifies the command name associated with this subcommand.
- The filetype for messages is HELPMSG.
- The filetype for a list of supported commands for a given function is HELPMENU.
- The filetype for a list of tasks supported for a given function is HELPTASK.
- The filetype for a list of abbreviations and synonyms for a given function is HELPABBR.

If the component name is shorter than four characters, the filetype is shortened. For example, HELPCP is the filetype for CP commands. If the component name is longer than four characters, only the first four characters are used. For example, HELPDEBU is the filetype for DEBUG subcommands.

The only exception to the above rule is for EXEC 2 HELP files. Because EXEC and EXEC 2 have the same first four characters, CMS examines the fifth character to determine if the request is for EXEC or EXEC 2. Since filetypes are limited to eight characters, CMS assigns the filetype HELPEXEC to EXEC files and the filetype HELPEXC2 to EXEC 2 files.

If you create a file with a fileid of HELP HELPMENU, the HELP facility will display your HELP HELPMENU file instead of the HELP HELPTASK file when HELP is invoked without any other parameters.

There are also certain filetypes that are reserved for the various components of the HELP facility:

Filetype	Reserved for
HELPABBR	Lists of command or subcommand abbreviations or synonyms for a component
HELPCP	CP commands
HELPCMS	CMS commands
HELPDEBU	DEBUG subcommands
HELPEDIT	EDIT subcommands
HELPEXEC	EXEC statements
HELPEXC2	EXEC 2 statements
HELPGROU	HELP files for the GROUP command

HELPHELP	HELP files for HELP
HELPIPCS	IPCS commands
HELPMENU	Menus of HELP components
HELPMSG	CP, CMS, GCS, IPCS, and TSAF messages
HELPQUER	XEDIT QUERY subcommands
HELPSET	XEDIT SET subcommands
HELPXEDI	XEDIT subcommands
HELPPREF	XEDIT PREFIX subcommands
HELPSQLD	SQL/Data System (5748-XXJ) Program Product (only if you have this installed on your system.)
HELPSRPI	Server-Requester Programming Interface subcommands
HELPREXX	System Product Interpreter Statements
HELPTASK	HELP task menus
HELPTSAF	Transparent Services Access Facility commands
HELPCMSQ	CMS QUERY operands
HELPCMSS	CMS SET operands
HELPCPQU	CP QUERY operands
HELPCPSE	CP SET operands

Examples of Naming Conventions

The following examples illustrate the naming conventions you would use to create specific files to work with the HELP command:

Filename	Filetype	Description
ACCESS	HELPCMS	A CMS command description
BEGIN	HELPCP	A CP command description

Filename	Filetype	Description
ADD	HELPXEDI	An XEDIT subcommand description
DMS186W	HELPMSG	A CMS message description
CMS	HELPMENU	A list of the CMS command and/or EXEC names supported by the HELP facility
HELP	HELPTASK	A list of HELP tasks supported by the HELP facility.
CMS	HELPABBR	A list of CMS abbreviations and synonyms supported by the HELP facility.

Creating Menus for HELP Files

The HELP facility has two types of menus:

1. Menus that are command component menus, having a filetype of HELPMENU

2. Menus that are task menus, having a filetype of HELPTASK.

The filename of a HELPMENU file is the name of the component to which the commands in the menu belong. For example, EXEC2 HELPMENU is the filename and filetype for the menu containing EXEC 2 statements. Menus with a filetype of HELPMENU contain a list of the HELP files for that component.

The filename of a HELPTASK file can be any filename up to eight characters and should describe the list of tasks contained in the file. For example, DISK HELPTASK is the filename and filetype for the task menu containing tasks which involve using disks. Menus with a filetype of HELPTASK contain a list of tasks.

Creating HELPMENU Files

There are a few rules you must follow when creating HELPMENU files:

1. Precede the list of names with any amount of information for the user. This information should include instructions for selecting an entry from the menu.

2. Include two blank lines between the information for the user and the list of names. If two consecutive blank lines are not found, then the file is considered pre-formatted and is not sorted or formatted by HELP.

Any two or more consecutive blank lines indicate the end of the user information section and the beginning of the list of names for the HELP files for that component. Therefore, you are limited to one blank line between lines in the user information area.

3. Following these two blank lines, enter the filenames in any order. Filenames must have the following characteristics:

   - They must begin in column 1.

   - The filenames must be one to a line and limited to eight characters.

     The list of names is sorted in ascending alphabetical order (in columns 1 thru 9) and is formatted by the HELP facility for display on the screen.

   - Names of submenus and subtasks should be prefixed with an extra character, "*" (asterisk) or ":" (colon), allowing for a length of nine characters. If an entry on a menu is the name of another menu, the entry must be preceded by an "*" (asterisk). For example, on the CMS MENU, the REXX entry is preceded by an "*." This means that if you select the *REXX entry, HELP will display a menu of REXX commands.

     If an entry on a menu is the name of a task menu, the entry must be preceded by a ":" (colon). For example, on the COMMANDS MENU, the TASK entry is preceded by a ":." If you select the :TASK entry, HELP will display a TASK MENU of tasks you may wish to perform.

     The filename of the menu file determines the component id for items selected from a menu. Thus, the menu file REXX HELPMENU is a menu of REXX keywords and functions. For example, when the ABBREV entry is selected from the REXX HELPMENU display, HELP will look for the ABBREV HELPREXX file and display it.

     It is possible to break down a set of commands by function and to create a menu that is a subset of a command set. A menu that is a list of the REXX built-in functions might be called FUNCTION HELPMENU. When you select an entry such as ABBREV from the FUNCTION HELPMENU, the ABBREV HELPFUNC file will be displayed if it exists. This means that to have the ABBREV entry on both the REXX menu and the FUNCTION menu, you must have an ABBREV HELPFUNC file and an ABBREV HELPREXX file.

To simplify the creation of menus that are a subset of another component, an additional control word, .MT (Menu Type), is allowed in menu files. The .MT control word defines a component to override the component derived from the filename. Therefore, you could include the following in your FUNCTION HELPMENU file:

```
.MT REXX
```

This control word tells HELP to look for a HELPREXX file instead of a HELPFUNC file when you select an entry from the FUNCTION HELPMENU. The .MT control word is treated as a comment if it appears in other types of HELP files.

Example of HELPMENU File Creation

Assume you want to add HELP files concerning your internal system (System 5) procedures to the HELP facility. You would follow the procedure outlined below.

1. Choose the component name of SYS5 (System 5)

2. Create the HELP files for these procedures.

3. Following the rules given in "Filenames for HELP Files" on page 267, give each procedure file a filename and filetype. Thus, the file containing the information about CLASS8 (a class identifying the type of printing desired) would be named CLASS8 HELPSYS5.

4. Create a new file and give it a filename of SYS5 and a filetype of HELPMENU. This file will be your menu.

5. In the first few lines of the menu, type in any descriptive information you wish to appear when the menu is displayed. Skip two lines after this information and list the names of all the files you want to access from the menu.

Your menu should look similar to the example below.

```
A file may be selected for viewing by placing the cursor under any
character of the file you want to display and pressing the ENTER key.


CLASS8
CLASS7
CLASSO
CLASSC
MOUNT
DEMOUNT
.
.
.
```

Figure 71. Sample of HELPMENU File Creation

When you specify HELP SYS5 MENU, the HELP facility will alphabetize and columnize the filenames and display this file. You may then work with this menu as you would with any other HELP menu.

Creating HELPTASK Files

There are a few rules you must follow when creating HELPTASK files:

1. The file must start with a section of information for the user. This information should include instructions for selecting an entry from the task menu.

2. The file must include two blank lines between the information for the user and the list of names. Any two or more consecutive blank lines indicate the end of the user information section and the beginning of a list of tasks. Therefore, you are limited to one blank line between lines in the user information area.

3. Following the two blank lines is the task selection section. The lines in the selection section are divided vertically into two parts:

   a. The first part, which starts in column 1 and ends in column 24, contains the operands and options of the HELP command to be issued when the item is selected.

   b. The second part, starting in column 25, is the task description.

For example,

```
CMS PRINT (ALL          Print a file
|_______________________|_______
1                       25
```

When you select the task "Print a file," the HELP file for the CMS PRINT command is displayed just as if you had entered HELP CMS PRINT (ALL on the command line.

Example of HELPTASK File Creation

Assume you want to add HELP files for your internal system tasks to the HELP facility. You would follow the procedure outlined below.

1. Choose the task name of INTERNAL (Internal Procedures).

2. Create the HELP files for these tasks.

3. Following the rules given in "Filenames for HELP Files" on page 267, give each task file a filename and filetype. Thus, the file containing the information about PRINT (printing a file on a local printer) would be named PRINT HELPINTE.

4. Create a new file and give it a filename of INTERNAL and a filetype of HELPTASK. This file will be your task menu.

5. In the first few lines of the task menu, type in any descriptive information you wish to appear when the task menu is displayed. Skip

two lines after this information and list in columns 1 through 24 the filenames of the files created for Step 2 above. In the columns after 25, add a description of each task.

Your task menu should look similar to the example below.

```
A task may be selected for viewing by placing the cursor under any
character of the task you want to display and pressing the ENTER key.


PRINT                        Print a file on a local printer.
PHONE                        Display a phone list for my department.
.
.
.
```

Figure 72. Sample of HELPTASK File Creation

When you specify HELP INTERNAL TASK, the HELP facility will alphabetize and columnize the filenames and display this file. You may then work with this task menu as you would with any other HELP menu.

Creating Command HELP Files

When you request HELP for a specific command, there are several options available to display only a single subset of the information available for that command. These options are BRIEF, RELATED, DESCRIPT, FORMAT, PARMS, OPTIONS, NOTES, and ERRORS. In addition, the DETAIL option can be used to display one or more of the DESCRIPT, FORMAT, PARMS, OPTIONS, NOTES, or ERRORS sections. (See the *VM/SP CMS Command Reference* for information on the use of these options).

In creating a HELP file, you must identify the parts of the information which will correspond to each option. You would do this by using the .CS format word to identify the beginning and end of each section of HELP text. You would note the beginning of a section by specifying ".CS xxx ON," where 'xxx' is the corresponding keyword or number for that section. After the text for the section, you would specify ".CS xxx OFF."

For example, before the text for a BRIEF section, you would specify ".CS BRIEF ON" or ".CS 0 ON." At the end of the text, you would specify ".CS BRIEF OFF" or ".CS 0 OFF." The text included in this section would be displayed when you request BRIEF HELP for a command.

The following tables show the format of a HELP file. Remember, you may use the numbers shown in the table on the left or the keywords shown in the table on the right.

```
.CS 0 on                         .CS BRIEF on
Text for BRIEF option             Text for BRIEF option
.CS 0 off                        .CS BRIEF off
.CS 1 on                         .CS DESCRIPT on
Text for DESCRIPT option          Text for DESCRIPT option
.CS 1 off                        .CS DESCRIPT off
.CS 2 on                         .CS FORMAT on
Text for FORMAT option            Text for FORMAT option
.CS 2 off                        .CS FORMAT off
.CS 3 on                         .CS PARMS on
Text for PARMS option             Text for PARMS option
.CS 3 off                        .CS PARMS off
.CS 4 on                         .CS OPTIONS on
Text for OPTIONS option           Text for OPTIONS option
.CS 4 off                        .CS OPTIONS off
.CS 5 on                         .CS NOTES on
Text for NOTES option             Text for NOTES option
.CS 5 off                        .CS NOTES off
.CS 6 on                         .CS ERRORS on
Text for ERRORS option            Text for ERRORS option
.CS 6 off                        .CS ERRORS off
.CS 7 on                         .CS RELATED on
Text for RELATED option           Text for RELATED option
.CS 7 off                        .CS RELATED off
```

BRIEF HELP

BRIEF HELP provides a short description of a command, its basic syntax (without options), and an example. If you wish to follow the current VM/SP conventions, any BRIEF HELP sections you create would contain this information in a maximum of ten lines and 75 columns of text.

DETAIL HELP

DETAIL HELP presents a complete description of a command, the command format, an explanation of its parameters and options, usage notes, and error information. It consists of the DESCRIPT, FORMAT, PARMS, OPTIONS, NOTES, and ERRORS sections.

The DESCRIPT Section: The DESCRIPT section should contain a description of the command.

The FORMAT Section: The FORMAT section should include a diagram of the command syntax, including any operands or options. You can use the same notational conventions used in the VM/SP HELP file. These conventions are listed below.

- Use the less than (<) and greater than (>) symbols to denote choices where the user *must select one*. For example,

  ```
  <A>
  ```

means that the user *must* specify A.

```
<A>
<B>
<C>
```

means that the user *must* specify either A, B, or C.

- The use of parentheses, bars, and plus signs means that the user does not have to select any of the items. For example,

  ```
  (A)
  ```

  means that the user *may* specify A or may omit the field.

  ```
  + +
  |A|
  |B|
  |C|
  + +
  ```

  means that the user *may* specify A, B, or C, or may omit the field.

The PARMS Section: The PARMS section should include a description of the parameters of the command.

The OPTIONS Section: The OPTIONS section should include a description of the options available for the command.

The NOTES Section: The NOTES section should include any usage notes and examples you wish to include in the HELP file.

The ERRORS Section: The ERRORS section contains any responses related to the command, a one-line statement referring you to the *VM/SP CMS Command Reference*, a one-line statement referring you to the *VM/SP System Messages and Codes* manual, and one sentence on how to use HELP to get information on a specific message issued by the command. If you wish to add additional information to your own HELP files, simply add the errors text that you want displayed.

RELATED HELP

RELATED HELP provides information on commands which are similar to a command presently being displayed. It is especially useful if you are looking for similar functions or tasks but are unsure of the specific command name.

The information in a RELATED section must be in task menu format. Refer to the section entitled "Creating HELPTASK Files" on page 274 for details. The following is an example of the format of the RELATED section for the ERASE command.

```
.cs 7 on
.cm (c) Copyright IBM Corporation 1986

For RELATED information on removing files or parts of files from your
virtual machine, place the cursor under the topic of your choice and
press ENTER or the PF1 key.

XEDIT DELETE (BRIEF      DELETE¢%   - Removes one or more lines from
XEDIT DELETE (BRIEF            ¢%     a file while using XEDIT.
XEDIT DELETE (BRIEF
DISCARD TASK             DISCARD¢%  - Removes files from your readerlist,
DISCARD TASK                   ¢%     filelist or PEEK screen.
DISCARD TASK
CMS ERASE (BRIEF         ERASE¢%    - Removes files from your minidisk.
CMS ERASE (BRIEF
CP PURGE (BRIEF          PURGE¢%    - Removes spool files from your
CP PURGE (BRIEF              ¢%       reader, printer or punch.
CP PURGE (BRIEF

.cs 7 off
```

Figure 73. Format of the RELATED Section of the ERASE Command

Highlighting Words Within a File

When you create and format your own HELP files, you may also wish to highlight certain words for emphasis. The HELP facility allows you to highlight selected portions of a file when those portions are viewed in display mode. To highlight a portion of a line, enclose that part of the line with the pairs of characters "¢|" and "¢%." The "¢|" control character turns the highlighting on (white on a color terminal); the "¢%" control character turns the highlighting off. Everything on the line appears highlighted until the "¢%" control characters are encountered. Care must be taken when inserting the highlight control characters within a HELP file because the control characters appear as a single blank when displayed on the screen (or typed on a line-oriented terminal). This may affect the formatting of a file.

Note: With TASK menus or RELATED sections, the selection entries are automatically displayed by HELP as highlighted lines. If you desire to highlight only a portion of these lines, the control characters "¢%" should be used to turn highlighting off at the desired place in the file. You can use these characters as explained above to turn highlighting on or off elsewhere in the line. The previous example of a RELATED section shows how "¢%" is used to turn highlighting off after selection entries.

Using Command Abbreviations

To be able to use the shortened version or a synonym of a command name in your HELP files, you must create an abbreviation file with the proper entry. This file has the name of the component to which the command belongs as a filename and and the filetype of HELPABBR. The abbreviation file contains the command, its synonym, or the proper abbreviation for the command name, and the number that represents the shortest possible character string that is acceptable. If the number is omitted, the default is the length of the second word. The command should be in upper case. Here is an example of an entry in an abbreviation file:

```
ACCESS    ACcess    2
```

Any of the following will invoke the CMS ACCESS command HELP file:

```
HELP AC
HELP ACC
HELP ACCE
HELP ACCES
HELP ACCESS
```

An abbreviation file may also be used to provide help for command synonyms and for command or subcommand names that contain special characters other than those supported by the HELP facility.

You should create an abbreviation file whenever a component is added to the HELP facility. The file should be given a filemode number of 2 (for example, CMS HELPABBR A2).

If no abbreviations are supported for the component, create an abbreviation file containing a comment line, indicated by an asterisk (*) in column 1. This will allow HELP to search for your HELP files more quickly.

Adding Your Own HELP Components

You can create your own HELP components for commands or applications you have added to your system. First, you must decide on the name for the component and create the HELP files you wish to be displayed for that component.

To display these HELP files, the user will be able to simply enter HELP 'componentname', followed by the name of one of your commands. This will invoke the HELP facility. The file with the filename 'command name' and filetype of HELPxxxx, where 'xxxx' is the first four characters of the component name, will be displayed.

You can also create a menu of your commands. See the section entitled "Creating HELP Files" on page 267 for details.

If you have used the parsing facility and defined your command syntax using the Definition Language for Command Syntax (DLCS), you must also update the APPLID HELPABBR file. This file is used by HELP to resolve synonyms, translations, and abbreviations for commands with syntax defined by using DLCS. When a user requests HELP for a command in your component, HELP uses the APPLID HELPABBREV file to locate the correct DLCS file application identifier.

For example, if you have created an application for generating monthly activity reports for your department called the Monthly Activity System, with a DLCS application identifier of MAS and a HELP component of MONTHLY, you should add the following line to the APPLID HELPABBR file:

```
MAS         MONTHLY
```

This will tell HELP that when a user enters the command HELP MONTHLY PRI, HELP should search the MAS DLCS file and recognize that PRI is an abbreviation for the PRINT command in your application. HELP will then display your PRINT HELPMONT HELP file.

Note: The APPLID HELPABBR file follows the same HELPABBR file format. This means that you could specify an abbreviation for your component. The length of the abbreviation for a component in the APPLID file must always be at least four characters for HELP to find the component HELP files.

Refer to *VM/SP CMS for System Programming* for more information on using the parsing facility and defining command syntax using DLCS.

Using HELPCONV to Create HELP Files

HELPCONV is an additional text processing tool which can help you to format your HELP files. HELPCONV format words can do the following:

- Draw boxes to enclose tables, illustrations, or text
- Place comments within a file
- Separate the sections of a HELP file
- Cause concatenation of input lines and left- and right-justification of output
- Indent only the next input line the specified number of spaces
- Indent a series of input lines the specified number of spaces
- Indent the specified number of spaces for all but the first line in a series of input lines
- Override the derived component for a menu file

- Insert blank lines between output lines
- Change the final output representation of any input character.

The HELPCONV command converts an unformatted HELP file containing SCRIPT format words into a formatted HELP file. These format words are summarized in Figure 74 on page 282. The following format words remain in the file:

```
.CS
.CM
.MT
```

The output file has the filetype $HLPxxxx, where 'xxxx' represents the last four characters of the filetype of the input file.

FORMAT WORD	OPERAND FORMAT	FUNCTION	BREAK	DEFAULT VALUE
.BX (BOX)	V1 V2...Vn OFF	Draws horizontal and vertical lines in the blank columns around subsequent output text.	Yes	Draws a horizontal line.
.CM (COMMENT)	Comments	Places comments in a file for future reference.	Yes	
.CS (CONDITIONAL SECTION)	n ON/OFF keyword ON/OFF	Separates sections of HELP files.	Yes	
.FO (FORMAT MODE)	ON/OFF	Causes concatenation of input lines, and left and right justification of output.	Yes	On
.IL (INDENT LINE)	n|+n|-n	Indents only the next line the specified number of spaces.	Yes	0
.IN (INDENT)	n|+n|-n	Specifies the number of spaces subsequent text is to be indented.	Yes	0
.MT (MENU TYPE)	component	Correlates a component to a menu file when the component is not to be derived from the filename. For files other than menu files, .MT is seen as a comment.	Yes	
.OF (OFFSET)	n|+n|-n	Provides a technique for indenting all but the first line of a section.	Yes	0
.SP (SPACE)	n	Specifies the number of blank lines to be inserted before the next output line.	Yes	1
.TR (TRANSLATE)	s t	Specifies the final output representation of any input character.	No	

Figure 74. HELP and HELPCONV Format Word Summary

.BX (BOX)

The HELPCONV command can insert vertical and horizontal lines in the formatted output to enclose text, illustrations, or tables. The BOX format word defines and initializes a horizontal rule for output and defines vertical rules for subsequent output lines.

Now for some examples. The first time you issue the .BX format word, specify the columns in which you want the vertical lines to appear. The highest value that may be specified for a column is 239. Issuing

```
.bx 1 10 20 30
```

results in the following output:

```
+--------+---------+---------+
```

Subsequently, entering the .BX format word with no operands causes HELPCONV to create a horizontal line with vertical bars at the columns indicated.

As HELPCONV formats each line, vertical bars are places in the columns that you specified on .BX, unless a column is already occupied by a data character. In this case, HELPCONV does not place a vertical bar in the column.

The next example shows how you can change the vertical structure several times in succession. Issuing

```
.bx 10 20
.sp
.bx 5 25
.sp
.bx 10 20
.sp
.bx 5 25
.sp
.bx 10 20
.sp
.bx off
```

results in:

```
      +---------+
      |         |
 +----+---------+----+
 |                   |
 +----+---------+----+
      |         |
 +----+---------+----+
 |                   |
 +----+---------+----+
      |         |
      +---------+
```

You can specify a .BX format word with different columns while a box is being drawn. When this happens, HELPCONV puts in vertical ascenders for all the old columns and vertical descenders for all the new columns. The vertical rules then appear in all subsequent output lines in the new columns designated.

The .BX format word causes a break in the text.

The column specification for the .BX format word uses a different rule than is used elsewhere in HELPCONV. In some control words, the numbers in the format word do not represent columns, but displacements.

For example, the HELPCONV format word .IN 5 means that a blank character should be expanded to enough blanks to fill up *through* column 5; the next word starts in column 6. In the .BX control word, .BX 5 means to put vertical rules *in* column 5. Thus, you can use the same numbers for a .IN control word as for a .BX control word, and the vertical bar will appear in the column immediately preceding the first word on that line.

For example, consider the file called MARYHADA that looks like this:

```
.fo off
.bx 5 43
.in 5
Mary had a little lamb,
Whose fleece was white as snow,
And everywhere that Mary went,
The lamb was sure to go.
.bx off
```

This file, when processed by HELPCONV, creates the following output:

```
Mary had a little lamb,
Whose fleece was white as snow,
And everywhere that Mary went,
The lamb was sure to go.
```

.CM (COMMENT)

Use the COMMENT format word to place comments within a HELP file. Comments are useful for:

- Tracking files.

  This is a useful way for you to keep track of your changes to HELP files. You can include a comment that gives your name, the date and reason you created a file, and a future date at which the file may be erased. HELP does not display any lines in a HELP file beginning with .CM. Therefore, you can include information about any alterations you have made to your HELP files in the files themselves.

- Documenting formats.

  If you use a special format in a HELP file that may be accessed by other people, you may want to place notes within the file explaining how to update the file.

- Place-holders.

If a file is incomplete, you may want to put comments in the file where information should be added later.

For example, you could add the following to your HELP file:

```
.CM Remember to change the date.
```

You would only see the line above when you were editing the HELP file.

You can also use comments to store unique identifications to be used to locate a specific region of the file during editing.

Comments cause a format break. Comment lines are retained in the formatted file and do not appear when the HELP file is displayed. They are not removed by the HELPCONV command.

.CS (CONDITIONAL SECTION)

The CONDITIONAL SECTION format word was explained in the section entitled "Creating Command HELP Files" on page 275. As we discussed earlier, the .CS format word enables you to identify the specific sections of the input file that are directly associated with the HELP facility command "options." These options are BRIEF, RELATED, DESCRIPT, FORMAT, PARMS, OPTIONS, NOTES, and ERRORS. (See the *VM/SP CMS Command Reference* for further information on the use of these options).

In order for HELP command processing to display the appropriate information, the format word ".CS n ON" or ".CS keyword ON" is required before each section of the HELP text. The format word ".CS n OFF" or ".CS keyword OFF" should follow each HELP section. This statement tells HELP that the end of the section has been reached.

The keyword designating each section corresponds exactly to the HELP option which specifies the section to be displayed. These keywords cannot be abbreviated in the HELP file.

The .CS format word acts as a break. If blank lines or portions of a file are between the conditional sections (.CS), these lines will be displayed with the DETAIL information. These conditional section lines are not removed from formatted output by the HELPCONV command.

.FO (FORMAT MODE)

Use the FORMAT MODE format word to cancel or to restore concatenation of input lines and right-justification of output lines.

Use .FO ON to restore default HELP formatting, including both justification and concatenation of lines. If you use the .FO format word with no operands, ON is assumed. Use .FO OFF to cancel concatenation of input lines and justification of output lines. Subsequent text is printed "as is."

When format mode is in effect, lines are formed by shifting words to or from the next line (concatenation) and by padding with extra blanks to produce an aligned right margin (justification).

The .FO format word acts as a break. When format mode is in effect, a line without any blanks that exceeds the current line length is extended into the right margin. If a line is processed so that only one word fits on the line, the word is left-justified.

If *no* formatting is to be done by HELPCONV, HELP description files *must* contain a .FO OFF format word before the section of text you wish to leave unformatted. .FO OFF should be placed as the first line of MENUS and TASK menus and before the RELATED section of a file.

For example, when .FO OFF is found, justification and concatenation are completed for the preceding line or lines, but the following lines are typed exactly as they appear in the file.

When .FO ON is found, justification and formatting are resumed with the next input line. Output from this point on in the file is padded to produce an aligned right margin on the output page.

Note: You may not want this type of formatting in all cases; you may want certain output to appear exactly as it appears in your file or when presented on the prior releases of VM/SP CMS HELP. In this case, place .FO OFF in the file.

.IL (INDENT LINE)

Use the INDENT LINE format word to set off paragraphs or portions of text by indenting them. This often improves the readability by emphasizing certain text. The INDENT LINE format word indents *only the next line* a specified number of characters. The line is shifted to the right or left of the current margin (which includes any indent or offset values in effect).

When successive .IL format words are encountered without intervening text, or when you specify positive or negative increments for .IL format words entered without intervening text, the indent amount is modified to reflect the last .IL encountered; that is, the increments are added together. Thus the lines

```
.il 4
.il +6
```

result in the next line being indented 10 spaces.

A negative value following .IL shifts the text to the left. If the resulting amount would cause a shift to the left of character position one, an error message is generated.

The .IL format word acts as a break. Therefore, text accumulated before the .IL format word is processed and displayed before the next piece of text is processed.

Since the .IL format word causes a break in text, you may find it useful to indicate the beginning of a new paragraph. For example:

```
.il 3
This line begins a new paragraph.
.il 3
This line begins another.
```

These lines result in:

```
   This line begins
a new paragraph.
   This line begins
another.
```

.IN (INDENT)

Use the INDENT format word to change the left margin displacement of HELP output.

For example,

```
This line is not indented.
.in 5
This line is indented.
```

results in:

```
This line is not indented.
     This line is indented.
```

The .IN format word resets the current left margin. For example, .IN 5 sets the left margin at 6, leaving 5 blank spaces at the left. This indentation remains in effect for all following lines until another .IN format word is encountered. .IN 0 cancels the indentation, and output continues at the original left margin setting.

.IN cancels any .OF (OFFSET) setting. The .OF 0 request cancels the current offset, but leaves the left margin specified by the .IN format word unchanged.

Since .IN causes a break, text accumulated before the .IN format word is processed and displayed, then the next text is processed.

The .IN format word effectively sets a new left margin for output text so that when you want text indented you do not have to enter blanks in front of the input lines (as you would for normal typing). HELPCONV continues to concatenate and justify input text lines that begin in column 1, but displays the output indented the number of spaces you specify.

Here's another example:

```
These few lines of text
are formatted
with enough words
.in 5
so that you can
see how HELP's formatting
process
.in +3
continues and may
.in -6
even be reversed, by using a
negative value.
```

These lines result in:

```
These  few  lines  of
text  are   formatted
with enough words
     so that  you can
     see  how  HELP's
     formatting
     process
        continues and
        may
  even  be  reversed,
  by using a negative
  value.
```

In this example, the first .IN format word (.in 5) shifts output to the right five spaces so that text begins in column 6. The second .IN format word (.in +3) requests that the current indentation increase by three spaces so the left margin is now in column 9. When you supply a negative value with the .IN format word (.in -6), the margin is shifted to the left.

.MT (MENU TYPE)

Use the .MT format word to specify the component of a menu. The .MT format word will override the default component of a menu file. When a menu file is used, the filename of the menu is used to generate the name of the component. This component is used to locate the appropriate HELP file when a selection is made. For example, if you select a command from the XEDIT menu, it is equivalent to issuing "HELP XEDIT command." If the line .MT xxxxx was included in the file, selecting a command from the menu would be equivalent to issuing "HELP XXXXX command."

The .MT format word is used to assist in the creation of menus that are a subset of another menu. For example, a menu that contains a list of REXX functions might be called FUNCTION HELPMENU. In this case, the HELP files for the individual functions can only be located if they are duplicated under the filetype HELPFUNC. The .MT control word defines a component id to override that derived from the filename. The FUNCTION menu could include:

```
.MT REXX
```

This specifies that the menu contains a list of REXX commands and thus will be found under the filetype HELPREXX.

The MENU TYPE format word acts as a break.

.OF (OFFSET)

Use the OFFSET format word to indent all but the first line of a block of text. An offset differs from an indentation. Offsets do not affect the first line immediately following the format word; the second and subsequent input lines are indented the specified number of characters. This is useful when formatting numbered lists where text is blocked to the right of the number.

The .OF format word does not take effect until after the next line is formatted. The indentation remains in effect until a .IN (INDENT) format word or another OFFSET control word is encountered.

You can use the .OF format word within a section that is also indented with the .IN format word. Note that .IN settings take precedence over .OF, however, any .IN request causes a previous offset to be cleared.

If you want to start a new section with the same offset as the previous section, you need only repeat the .OF n request.

This format word acts as a break; subsequent text is printed at the current left margin, that is, whatever the indentation is (0, if no .IN format word is in effect).

.OF shifts all but the first line of text. You can use the .IL (INDENT LINE) format word to shift only the next line to the left or right of the current margin.

Following is an example to try:

1. Starting an offset:

   ```
   .of 10
   The line immediately following the .OF format word is
             printed at the current left margin.  All lines
             thereafter (until the next indent or offset
             request) are indented ten spaces from the
             current margin setting.  These two examples
             were processed with OFFSET control words in
             the positions shown.
   ```

2. Ending an offset:

   ```
   .of
   ```

The effect of any previous .OF request is canceled, and all output after the next line continues at the current left margin setting.

.SP (SPACE LINES)

Use the SPACE LINES format word when you want blank lines to appear between text lines of output.

For example:

```
.sp 5
```

indicates that you want to leave five lines of space in the text output. You can use multiple spaces when you want a heading or a title to stand out, for example the lines:

```
A Love Story
.sp 5
The quick brown fox
was eager
to meet the pretty poodle.
```

will result in:

```
A Love Story





The  quick  brown  fox
was eager to  meet the
pretty poodle.
```

.TR (TRANSLATE CHARACTER)

The TRANSLATE CHARACTER format word allows you to specify the output representation of each character in the source text. For example, you could specify that all blanks in the file appear as question marks in the output.

After formatting of an input source line has been completed and immediately before actual output, each character of the output line may be translated to a different output code.

Translate character specifications remain in effect until explicitly respecified. Since format words are only processed internally, they are never translated in the file.

A .TR format word with no operands causes the translation table to be reinitialized and all previously specified translations to be reset.

The .TR format word does not cause a break. If you have a section of text that has translation characters in effect, followed by a .TR to reset the translations, the last line of the text may not yet have been printed. In this case, that last line is not translated.

For example:

```
.tr 40 ?
```

This causes all blanks in the file to be typed as question marks (?) on output.

Changing Existing HELP Files

Because all HELP files are CMS files, you can add or delete files or menus or change any existing file or menu. However, there are a few restrictions you must follow when tailoring HELP files; they are discussed in the following sections.

Note: If you tailor your HELP files, you should retain documentation of the changes you've made by using the .CM format word to indicate that what follows is a comment. You can use this documentation later to help you update your files when IBM issues updates to the HELP facility files.

Adding HELP Files

The HELP facility enables the user to:

- Add HELP files to existing components or create a new component with its own HELP files
- Modify the command and message description files IBM provides with additional description files of the user's choice
- Produce a formatted HELP file by using the HELPCONV command and the HELP format words when creating the HELP description file.

  To create your own HELP file, follow the instructions in the section entitled "Creating HELP Files" on page 267.

Note: If you add HELP files to an existing component, you should follow the naming conventions for HELP files given in this chapter. If you update a component, you should also update its menu. For information on menus refer to "Creating Menus for HELP Files" on page 271.

A file that contains control words other than .CM, .CS, .FO, or .MT and has not been processed by the HELPCONV command will be identified by HELP as being unformatted or containing extra format words.

Deleting HELP Files

You delete HELP files just as you delete any CMS file, by specifying ERASE filename filetype. If you delete a file, you should delete the filename from the menu for that component and also from any task file.

Altering Existing HELP Files

To alter a HELP file:

1. Edit the file with a text processing editor

2. Add or delete as you wish, making sure that you follow the instructions given in "Creating HELP Files" on page 267.

3. Put .FO OFF after the informational paragraph in the RELATED sections prior to using HELPCONV to format the file.

If your installation maintains several releases of VM/SP CMS, and runs the HELP facility on them, you can modify the VM/SP CMS Release 4.0 (and following releases) formatted informational files so that they appear the same on all the releases. You would do this by adding the .FO OFF format word as the first statement of the HELP file and changing any unsupported format words to comments. For example, to allow a pre-formatted file to be used on VM/SP CMS Release 3.0, place .FO OFF before the informational HELP text lines. The .FO OFF format word tells HELP not to format the files but to display the information in its present format.

You must also change any .CS keyword ON/OFF lines to use the section number instead of the keyword. For example, .CS DESCRIPT ON would become .CS 1 ON. The .CS lines for the BRIEF and RELATED sections must also be changed to .CM, or the text for those sections must be included in a different section.

Note: You cannot use HELPCONV on a filetype of HELPTASK or HELPMENU unless the format word .FO OFF appears after the informational paragraph in the file.

Changing Menus

If you add, delete or change files, you must change the associated menu. Edit the menu file (the filename is component name; the filetype is HELPMENU) and make the necessary changes. Remember, there is an eight-character limit on filenames (a nine-character limit for submenus and subtasks). Only one filename goes on a line, and you can insert filenames anywhere in the list. If you delete a HELP file, you should delete from all HELPMENU files any line on which the filename occurs.

Appendixes

- Appendix A, "Considerations for Line Mode Terminals" on page 295
- Appendix B, "Summary of CMS Commands" on page 299
- Appendix C, "Summary of CP Commands" on page 311
- Appendix D, "Considerations for Full-Screen CMS and Windowing" on page 321

Appendix A. Considerations for Line Mode Terminals

Logical Line Editing Symbols

To aid you in entering command or data lines from your terminal, VM/SP provides a set of logical line editing symbols, which you can use to correct mistakes as you enter lines. Each symbol has been assigned a default character value. These normally are:

Symbol	Character
Logical character delete	@
Logical line end	#
Logical line delete	¢
Logical escape	"

Logical Character Delete

The logical character delete symbol (@) allows you to delete one or more of the previous characters entered. The @ deletes one character per @ entered, including the ¢ and # logical editing characters. For example:

```
ABC#@@ results in AB
ABC@D results in ABD
¢@DEF results in DEF
ABC@@@ deletes the entire string
```

Logical Line End

The logical line end symbol (#) allows you to key in more than one command on the same line, and thus minimizes the amount of time you have to wait between entering commands. You type the # at the end of each logical command line, and follow it with the next logical command line. VM/SP stacks the commands and executes them in sequence. For example, the entry:

```
query blip#query rdymsg#query search
```

is executed in the same way as the entries:

```
query blip
query rdymsg
query search
```

The logical line end symbol also has special significance for the #CP function. Beginning any physical line with #CP indicates that you are entering a command that is to be processed by CP immediately. If you have set a character other than # as your logical line end symbol, you should use that character instead of a #.

Logical Line Delete

The logical line delete symbol (¢) deletes the entire previous physical line, or the last logical line back to (and including) the previous logical line end (#). You can use it to cancel a line containing many or serious errors. If a # immediately precedes the ¢ sign, only the # sign is deleted, since the # indicates the beginning of a new line, and the ¢ cancels the current line. For example:

- Logical Line Delete:

  ```
  ABC#DEF¢ deletes the #DEF and results in ABC
  ABC#¢ results in ABC
  ABC#DEF¢#GHI results in ABC#GHI
  ABC#DEF¢GHI results in ABCGHI
  ```

- Physical Line Delete:

  ```
  ABC¢ deletes the whole line
  ```

Note: When you cancel a line by using the ¢ logical line delete symbol, you do not need to press a carriage return; you can continue entering data on the same line.

Logical Escape

The logical escape symbol (") causes VM/SP to consider the next character entered to be a data character, even if it is normally one of the logical line editing symbols (@, ¢, ", or #). For example:

```
ABC"¢D results in ABC¢D
""ABC"" results in "ABC"
```

If you enter a single logical escape symbol (") as the last character on a line, or on a line by itself, it is ignored.

When you enter logical escape characters in conjunction with other logical editing characters, the results may be difficult to predict. For example, the lines:

```
ABC""@DEF
ABC""@@DEF
```

both result in the line ABCDEF.

Defining Logical Line Editing Symbols

The logical line editing symbols are defined for each virtual machine during VM/SP system generation. If your terminal's keyboard lacks any of these special characters, your installation can define other special characters for logical line editing. You can find out what logical line editing symbols are in effect for your virtual machine by entering the command:

```
[illegible]
```

The response might be something like:

```
LINEND # , LINEDEL ¢ , CHARDEL @ , ESCAPE "
LINESIZE 130, MASK OFF, APL OFF, ATTN OFF, MODE VM
```

You can use the CP TERMINAL command to change the logical line editing characters for your virtual machine. For example, if you enter:

```
[illegible]
```

Then, the line:

```
[illegible]
```

would be interpreted:

```
input # line
input
#
```

The terminal characteristics listed in the response to the CP QUERY TERMINAL command are all controlled by operands of the CP TERMINAL command.

Appendix B. Summary of CMS Commands

Figure 75 on page 302 and Figure 76 on page 308 contain alphabetical lists of the CMS commands and the functions performed by each. Figure 75 on page 302 lists those commands that are available for general use; Figure 76 on page 308 lists the commands used by system programmers and system support personnel who are responsible for generating, maintaining, and updating VM/SP. Unless otherwise noted, CMS commands are described in *VM/SP CMS Command Reference.* If information on a particular command is found in another book, the listing in Figure 75 on page 302 or Figure 76 on page 308 will contain one of the following codes:

Code	Meaning
VSE PP	Indicates that this command invokes a VSE Program Product, available from IBM for a license fee.
EREP	Indicates that this command is described in *OS/VS Environmental Recording Editing and Printing (EREP) Program.*
IOCP UG	Indicates that this command is described in the *Input/Output Configuration Program User's Guide and Reference.*
DIAG	Indicates that this command is described in *VM Diagnosis Guide.*
OS PP	Indicates that this command invokes an OS program product, available from IBM for a license fee.
PLNGDE	Indicates that this command is described in the *VM/SP Planning Guide and Reference.*
INST	Indicates that this command is described in the *VM/SP Installation Guide.*
SFPROG	Indicates that this command is described in *VM System Facilities for Programming.*
CMSPROG	Indicates that this command is described in *VM/SP CMS for System Programming.*
CPPROG	Indicates that this command is described in *VM/SP CP for System Programming.*

There are ten commands called Immediate commands that are handled in a different manner from the other commands listed in Figure 75 and Figure 76. They may be entered while another command is executing by pressing the Attention key (or its equivalent) and are executed immediately.

The Immediate commands are:

HB	Halt batch execution
HI	Halt Interpretation
HO	Halt tracing
HT	Halt typing
HX	Halt execution
RO	Resume tracing
RT	Resume typing
SO	Suspend tracing
TE	Trace end
TS	Trace start

You can define your own Immediate commands by using any of the following:

- The IMMCMD macro in an assembler language program
- The IMMCMD command within an EXEC (CMS EXEC, EXEC 2, System Product Interpreter)
- NUCXLOAD command with the IMMCMD option specified.

Border commands are single-character windowing commands that you may enter in the corners of window borders. Following is a list of Border commands:

B	Scrolls the window backward
C	Clears the window of scrollable data
D	Drops the window
F	Scrolls the window forward
H	Hides the window
L	Scrolls the window to the left
M	Changes the location of the window

N	Minimizes the window
O	Restores the window
P	Pops the window
R	Scrolls the window to the right
S	Changes the size of the window
X	Maximizes the window

Command	Code	Usage
ACCESS		Identify direct access space to a CMS virtual machine, create extensions and relate the disk space to a logical directory.
ALARM VSCREEN		Sound the terminal alarm the next time the display is refreshed.
AMSERV		Invoke access method services utility functions to create, alter, list, copy, delete, import, or export VSAM catalogs and data sets.
ASSEMBLE		Assemble assembler language source code.
ASSGN		Assign or unassign a CMS/DOS system or programmer logical unit for a virtual I/O device.
CATCHECK		Allows a CMS VSAM user (with or without DOS set ON) to invoke the VSE/VSAM Catalog Check Service Aid to verify a complete catalog structure.
CLEAR VSCREEN		Erase data in the virtual screen by overwriting the data buffer with nulls.
CLEAR WINDOW		Scroll past all data in the virtual screen to which the window is connected so that no scrollable data is displayed in the window.
CMDCALL		Convert EXEC 2 extended plist function calls to CMS extended plist command calls.
CMSBATCH		Invoke the CMS batch facility.
CMSSERV		Start Enhanced Connectivity Facilities communications between your VM/SP host system and your work station (IBM Personal Computer).
COMPARE		Compare records in CMS disk files.
CONVERT COMMANDS		Convert a CMS file containing Definition Language for Command Syntax (DLCS) statements into an internal form for the parsing facility.
CONWAIT		Causes a program to wait until all pending terminal I/O is complete.
COPYFILE		Copy CMS disk files according to specifications.
CP		Enter CP commands from the CMS environment.
CURSOR VSCREEN		Position the cursor on specified line and column in a virtual screen.
DDR		Perform backup, restore, and copy operations for disks.
DEBUG		Enter DEBUG subcommand environment.
DEFAULTS		Set or display default options for the commands: FILELIST, NOTE, RDRLIST, RECEIVE, PEEK, SENDFILE, and TELL.
DEFINE VSCREEN		Create a virtual screen.
DEFINE WINDOW		Create a window.
DELETE VSCREEN		Remove a virtual screen definition.

Figure 75 (Part 1 of 6). CMS Command Summary

Command	Code	Usage
DELETE WINDOW		Remove a window definition.
DESBUF		Clears the program stack and the terminal input buffers.
DISK		Perform disk-to-card and card-to-disk operations for CMS files.
DLBL		Define a VSE filename or VSAM ddname and relate that name to a disk file.
DOSLIB		Delete, compact, or list information about the phases of a CMS/DOS phase library.
DOSLKED		Link-edit CMS text decks or object modules from a VSE relocatable library and place them in executable form in a CMS/DOS phase library.
DOSPLI	VSE PP	Compile DOS PL/I source code under CMS/DOS.
DROP WINDOW		Move a window down in the order of displayed windows.
DROPBUF		Eliminate a program stack buffer.
DSERV		Display information contained in the VSE core image, relocatable, source, procedure, and transient directories.
EDIT		Invoke the VM/SP System Product Editor in CMS editor (EDIT) compatibility mode to create or modify a disk file.
ERASE		Delete CMS disk files.
ESERV		Display, punch or print an edited (compressed) macro from a VSE source statement library (E sublibrary).
EXEC		Execute special procedures made up of frequently used sequences of commands.
EXECDROP		Purge storage-resident EXECs.
EXECIO		Do I/O operations between a device and the program stack or a variable.
EXECLOAD		Load EXECs into storage.
EXECMAP		List storage-resident EXECs.
EXECOS		Resets the OS and VSAM environments under CMS without returning to the interactive environment.
EXECSTAT		Obtain status of a specific EXEC.
EXECUPDT		Produces an updated executable version of a System Product Interpreter source program.
FCOBOL	VSE PP	Compile DOS/VS COBOL source code under CMS/DOS.
FETCH		Fetch a CMS/DOS or VSE executable phase.
FILEDEF		Define an OS ddname and relate that ddname to any device supported by CMS.

Figure 75 (Part 2 of 6). CMS Command Summary

Command	Code	Usage
FILELIST		List information about CMS disk files, with the ability to edit and issue commands from the list.
FINIS		Close an open file.
FORMAT		Prepare disks in CMS fixed block format.
GENDIRT		Fill in auxiliary module directories.
GENMOD		Generate nonrelocatable CMS files (MODULE files).
GENMSG		Convert a message repository file into an internal form.
GET VSCREEN		Write data from a CMS file to the specified virtual screen.
GLOBAL		Identify specific CMS libraries to be searched for macros, copy files, missing subroutines, LOADLIB modules, or DOS executable phases.
GLOBALV		Set, maintain, and retrieve a collection of named variables.
HELP		Display information about CP, CMS, or user commands, EDIT, XEDIT, or DEBUG subcommands, EXEC, EXEC 2 and System Product Interpreter control statements, and descriptions of CMS and CP messages.
HELPCONV		Convert a script file into an acceptable form to be used by the HELP facility.
HIDE WINDOW		Prevent the specified window from being displayed, and connect the window to a virtual screen.
IDENTIFY		Display or stack userid, nodeid, rscsid, date, time, time zone, and day of the week.
IMMCMD		Use the IMMCMD command to establish or cancel Immediate commands from within an EXEC.
INCLUDE		Bring additional TEXT files into storage and establish linkages.
IOCP	IOCP UG	Invoke the Input/Output Configuration Program
LABELDEF		Specify standard HDR1 and EOF1 tape label description information for CMS, CMS/DOS, and OS simulation.
LISTDS		List information about data sets and space allocation on OS, DOS, and VSAM disks.
LISTFILE		List information about CMS disk files.
LISTIO		Display information concerning CMS/DOS system and programmer logical units.
LKED		Link edit a CMS TEXT file or OS object module into a CMS LOADLIB.
LOAD		Bring TEXT files into storage for execution.
LOADLIB		Maintain CMS LOADLIB libraries.

Figure 75 (Part 3 of 6). CMS Command Summary

Command	Code	Usage
LOADMOD		Bring a single MODULE file into storage.
MACLIB		Create or modify CMS macro libraries.
MACLIST		List information about all members in a specified maclib, with the ability to edit and issue commands from the list.
MAKEBUF		Create a new program stack buffer.
MAXIMIZE WINDOW		Expand a window to the physical screen size.
MINIMIZE WINDOW		Reduce the size of the window to one line.
MODMAP		Display the load map of a MODULE file.
MOREHELP		Obtain either additional or related information about the latest valid HELP command you issued.
MOVEFILE		Move data from one device to another device of the same or a different type.
NAMEFIND		Display/stack information from a NAMES file. (default 'userid NAMES').
NAMES		Display a menu to create, display or modify entries in a 'userid NAMES' file. (The menu is available only on display terminals.)
NOTE		Prepare a 'note' for one or more computer users, to be sent via the SENDFILE command.
NUCXDROP		Delete specified nucleus extensions.
NUCXLOAD		Load a nucleus extension.
NUCXMAP		Identify existing nucleus extensions.
OPTION		Change the DOS/VS COBOL compiler (FCOBOL) options that are in effect for the current terminal session.
OSRUN		Load, relocate, and execute a load module from a CMS LOADLIB or OS module library.
PARSECMD		Call the parsing facility from within an EXEC.
PEEK		Display a file that is in your virtual reader without reading it onto disk.
POP WINDOW		Move a window up in the order of displayed windows.
POSITION WINDOW		Change the location of a window on the physical screen.
PRINT		Spool a specified CMS file to the virtual printer.
PSERV		Copy a procedure from the VSE procedure library onto a CMS disk, display the procedure at the terminal, or spool the procedure to the virtual punch or printer.
PUNCH		Spool a copy of a CMS file to the virtual punch.
PUT SCREEN		Make a copy of the physical screen and write the image to a CMS file.
PUT VSCREEN		Write the data from the scrollable data area of a virtual screen to a CMS file.

Figure 75 (Part 4 of 6). CMS Command Summary

Command	Code	Usage
QUERY		Request information about a CMS virtual machine.
RDR		Generate a return code and either display or stack a message that identifies the characteristics of the next file in your virtual reader.
RDRLIST		Display information about files in your virtual reader with the ability to issue commands from the list.
READCARD		Read data from spooled card input device.
RECEIVE		Read onto disk a file or note that is in your virtual reader.
REFRESH		Update virtual screens and their associated windows and refresh the screen.
RELEASE		Make a disk and its directory inaccessible to a CMS virtual machine.
RENAME		Change the name of a CMS file or files.
RESERVE		Use the RESERVE command to allocate all available blocks of a 512-, 1K-, 2K-, or 4K-byte block formatted minidisk to a unique CMS file.
RESTORE WINDOW		Return a maximized or minimized window to its size and location prior to the maximize or minimize.
ROUTE		Direct data of a particular message class to a virtual screen.
RSERV		Copy a VSE relocatable module onto a CMS disk, display it at the terminal, or spool a copy to the virtual punch or printer.
RUN		Initiate series of functions to be performed on a source, MODULE, TEXT, or EXEC file. SCRIPT control words in the document file.
SCROLL		Move a window to a new location on the virtual screen.
SENDFILE		Send files or notes to one or more computer users, attached locally or remotely, by issuing the command or by using a menu.(display terminal only)
SENTRIES		Determine the number of lines currently in the program stack.
SET		Establish, set, or reset CMS virtual machine characteristics.
SETPRT		Load a virtual 3800 printer.
SHOW WINDOW		Place a window on top of all other displayed windows and connect the window to a virtual screen.
SIZE WINDOW		Change the number of lines and columns for a specified window.
SORT		Arrange a specified file in ascending order according to sort fields in the data records.
SSERV		Copy a VSE source statement book onto a CMS disk, display it at the terminal, or spool a copy to the virtual punch or printer.

Figure 75 (Part 5 of 6). CMS Command Summary

Command	Code	Usage
START		Begin execution of programs previously loaded (OS and CMS) or fetched (CMS/DOS).
STATE		Verify the existence of a CMS disk file.
STATEW		Verify a file on a read/write CMS disk.
SVCTRACE		Record information about supervisor calls.
SYNONYM		Invoke a table containing synonyms you have created for CMS and user-written commands.
TAPE		Perform tape-to-disk and disk-to-tape operations for CMS files, position tapes, and display or write VOL1 labels.
TAPEMAC		Create CMS MACLIB libraries directly from an IEHMOVE-created partitioned data set on tape.
TAPPDS		Load OS partitioned data set (PDS) files or card image files from tape to disk.
TELL		Send a message to one or more computer users who are logged on to your computer or to one attached to yours via RSCS.
TXTLIB		Generate and modify text libraries.
TYPE		Display all or part of a CMS file at the terminal.
UPDATE		Make changes in a program source file as defined by control cards in a control file.
VALIDATE		Verify the syntax of a file identifier and verify whether or not a disk is accessed.
VSAPL	OS PP	Invoke VS APL interface in CMS.
WAITREAD VSCREEN		Use from an EXEC to update the virtual screen with data, refresh the physical screen, and wait for the next attention interrupt.
WAITT VSCREEN		Update the virtual screen with data.
WRITE VSCREEN		Enter information in a virtual screen.
XEDIT		Invoke the VM/SP System Product Editor to create or modify a disk file.
XMITMSG		Retrieve a message from a CMS message repository file or your own message repository file.

Figure 75 (Part 6 of 6). CMS Command Summary

Command	Code	Usage
ASM3705	INST	Assemble 370x source code.
ASMGEND	INST	Regenerate the VM/SP assembler command modules.
CMSGEND	INST	Generate a new CMS disk-resident module from updated TEXT files.
CPEREP	EREP	Format and edit system error records for output.
DCSSGEN	PLNGDE	Load, build, and save a DCSS containing the executing copy of frequently used EXECs and Macros.
DIRECT	PLNGDE	Set up VM/SP directory entries.
DISKMAP	INST	Summarize the MDISK statements in the CP directory to show gaps and overlaps in minidisk assignments.
DOSGEN	INST	Load and save CMSDOS and INSTVSAM shared segments.
DUMPSCAN	DIAG	Provide interactive analysis of CP abend dumps.
EXPAND	DIAG	Adds space to a program in object deck form.
GEN3705	INST	Generate an EXEC file that assembles and link-edits the 370x control program.
GENTSAF	INST	Builds the TSAF load module from TEXT files.
ITASK	INST	Performs most of the installation procedure by invoking other EXECs and commands.
LANGGEN	CMSPROG	Save all text files for a language in a DCSS, and/or save CP message repository.
LANGMERG	CMSPROG	Combine all language-related files for an application into one text file.
NCPDUMP	DIAG	Process CP spool reader files created by 370x dumping operations.
PRB	DIAG	Update IPCS problem status.
PROB	DIAG	Enter a problem report in IPCS.
PROP	SFPROG	Provide Programmable Operator capability.
SAMGEN	INST	Load and save the CMSBAM shared segment.
SAVENCP	INST, CPPROG	Read 370x control program load into virtual storage and save an image on a CP-owned disk.
SETKEY	CPPROG	Assign storage protect keys to storage assigned to named systems.
SNTMAP	INST	Processes DMKSNT macro definitions and produces a saved segment DASD map and a virtual memory map.
SPGEN	INST	Performs various system generation and maintenance functions.
SPLOAD	INST	Loads the VM/SP product tapes to the appropriate minidisks during initial installation.
STAT	DIAG	Display the status of reported system problems.
TRAPRED	DIAG	Allow the data collected by CPTRAP to be displayed or printed.

Figure 76 (Part 1 of 2). CMS Commands for System Programmers

Command	Code	Usage
UTILITY	INST	Provides installation functions such as printing system definition files, creating stand-alone service utility tape and service programs on disk, etc.
VMFASM	INST	Creates an updated source file using IBM updates, PTFs, and user updates, then assembles the source file.
VMFDOS	INST	Create CMS files for VSE modules from VSE library distribution tape or SYSIN tape.
VMFLKED	INST	Invokes the CMS LKED command to link-edit modules into a LOADLIB.
VMFLOAD	INST	Generate a new CP, CMS, or RSCS module.
VMFMAC	INST	Creates macro libraries using IBM and user updates.
VMFMERGE	INST	Applies PTFs to object code and maintains a record in the Merge Log.
VMFNLS	INST	Applies updates to national language files and compiles the updated versions.
VMFPLC2	INST	Loads source code from product tape, dumps CMS-formatted files from disk to tape, loads previously dumped files from tape to disk, performs various control operations on a specified tape drive, and loads the service installation VMSERV EXEC from the PUT.
VMFREMOV	INST	Removes PTFs that were applied using VMFMERGE.
VMFTXT	INST	Creates text libraries using IBM and user updates.
VMFZAP	INST	Applies ZAPs to object code and maintains a record of them in the ZAP Log.
VMSERV	INST	Controls the individual service EXECs on the system Program Update Tape.
VRSIZE	INST	Generates DMKSLC text which is used to generate a V=R area on the system.
VSAMGEN	INST	Load and save CMSVSAM and CMSAMS shared segments.
VSEVSAM	INST	Build a VSE/VSAM maclib containing the supported VSE/VSAM macros as well as the following VSE macros: CDLOAD, CLOSE, CLOSER,GET, OPEN, OPENR, and PUT.
ZAP	INST	Modify or dump LOADLIB, TXTLIB, or MODULE files.
ZAPTEXT	INST	Modifies or dumps individual text files.

Figure 76 (Part 2 of 2). CMS Commands for System Programmers

Appendix C. Summary of CP Commands

The CP commands are interactive console functions that you can use to control the VM/SP system. CP commands let you control your virtual machine. The commands that you can issue depend on your assigned privilege class(es), as described in this section. If appropriate, CP shows its processing results with responses (for example: COMMAND COMPLETE, MISSING ARGUMENT, or INVALID OPTION).

Privilege Classes for CP Commands

Each CP command has one or more of the following function types:

1. Operations
2. Resource
3. Programmer
4. Spooling
5. Analyst
6. CE (Customer Engineer - Service)
7. General.

The IBM-defined class structure is based on these seven function types. A command keeps its function type even if your installation establishes its own class structure. In most cases, each command class (A-G) has a corresponding function type (O,R,P,S,A,C or G). Some commands fall into more than one class. Figure 77 on page 312 shows the function of each privilege class and function type. Then, each CP command is listed in a table, along with its IBM-defined privilege class, function type, and a description of what each command does. If your installation changes the class for a command, record the change in the blank column in the summary table.

Your installation will assign each user, as part of the user's entry in the directory, one or more privilege classes. The exceptions are users with a password of NOLOG. These users have no privilege class and can only:

- Send messages
- Receive spooled output as punched cards or printed forms.

The NOLOG password identifies them to receive spooled output when a virtual machine user spools output for them.

The 'CP Class Any' commands are available to all user classes. These commands perform basic functions required by all virtual machines, such as

- Logging on
- Logging off
- Sending messages.

If a user tries to issue a command that does not have his command class, CP does not execute the command and issues an error message.

Note: If your installation adds or removes any commands from the general user class (IBM-defined class G), your installation should update the HELP files to show these changes. See Chapter 16, "Tailoring the HELP Facility" for details.

This table shows the different privilege classes, the function codes, and the major tasks that can be performed for each privilege class.

IBM-Defined Class	Function	Function, Primary User, and Use
A	O	**Operations** - *Primary system operator* The system assigns class A to the user at the VM/SP console during IPL. The class A user is responsible for VM/SP's availability and its communication lines and resources. These commands control system accounting, broadcast messages, run virtual machine performance options and affect VM/SP performance. Note: The Class A system operator who is automatically logged on during CP initialization is designated as the primary system operator.
B	R	**Resource** - *System Resource Operator* These commands control allocation and deallocation of real resources of the VM/SP system, except those that the primary system operator and the spooling operator control.
C	P	**Programming** - *System programmer* These commands update functions of the VM/SP system and change real storage in the real machine.
D	S	**Spooling** - *Spooling operator* These commands control spool data files and specific functions of the system's unit record equipment.
E	A	**Analyzing** - *System analyst* These commands examine and save certain data in the VM/SP storage area.

Figure 77 (Part 1 of 2). CP Privilege Classes

IBM-Defined Class	Function	Function, Primary User, and Use
F	C	**CE** -*Service Representative* (Customer Engineer) These commands get and examine data about input and output devices connected to the VM/SP system.
G	G	**General** - *General User* These commands control functions to run users' virtual machines.
Any	None	These CP commands are available to any user. These are to gain and take away access to the VM/SP system.

Figure 77 (Part 2 of 2). CP Privilege Classes

CP Commands

The following table lists the CP commands in alphabetical order, including:

- The IBM-defined privilege classes that can execute the command
- The corresponding function type
- A blank column for you to record the user-defined classes
- A brief description of each command.

Note: Brackets indicate that it is not necessary to specify the 'Function Type' in the OVERRIDE control statement when changing the IBM-defined privilege class. For more information on using the OVERRIDE control statement, refer to *VM/SP CP for System Programming.*

Command	Function Type	IBM-Defined Privilege Class	User-Defined Class	Usage
*	N/A	Any	None	Annotate the console sheet.
#CP	N/A	Any	None	Execute a CP command while remaining in the virtual machine environment.
ACNT	<O>	A		Create accounting records for logged on users, and reset accounting data. ACNT also closes the spool file that is accumulating accounting records.
ADSTOP	<G>	G		Halt execution at a specific virtual machine instruction address.

Figure 78 (Part 1 of 8). CP Command Summary

Command	Function Type	IBM-Defined Privilege Class	User-Defined Class	Usage
ATTACH	<R>	B		Logically connect a real device to a virtual machine for that machine's exclusive use or logically connect a DASD device for CP access and control. With CHANNEL operand, dedicate all devices on a particular channel to a specific user.
ATTN	<G>	G		Make an attention interruption pending for the virtual machine console.
AUTOLOG	<O>	A,B		Log on any virtual machine defined in the directory.
BACKSPAC	<S>	D		Restart or reposition the current output on a real punch or printer.
BEGIN	<G>	G		Continue or resume execution of the virtual machine at either a specific storage location or at the address in the current PSW.
CHANGE	S	D		Alter one or more external attributes of a closed spool file or files.
	G	G		Alter one or more attributes of a closed spool file.
CLOSE	<G>	G		Terminate spooling operations on a virtual card reader, punch, printer, or console.
COMMANDS	N/A	Any	None	Display the commands and diagnose codes you are authorized to use.
COUPLE	<G>	G		Connect channel-to-channel devices.
CP	N/A	Any	None	Execute a CP command while remaining in the CMS virtual machine environment.
CPTRAP	<P>	C		Create a reader file of selected trace table, CP interface, and virtual machine interface entries for problem determination.
DCP	<P>	C,E		Display the contents of real storage locations at the terminal.
DEFINE	R	B		Redefine the status of a 3330V volume.
	G	G		Reconfigure your virtual machine.

Figure 78 (Part 2 of 8). CP Command Summary

Command	Function Type	IBM-Defined Privilege Class	User-Defined Class	Usage
DETACH	R	B		Remove a real device from the CP system. With the CHANNEL operand, remove a dedicated channel from a user.
	G	G		Detach a virtual device from a virtual machine. Detach a channel from your virtual machine.
DIAL	N/A	Any	None	Connect a terminal or display device to the virtual machine's virtual communication line.
DISABLE	<R>	A, B		Prevent low-speed communication lines from accessing the system.
DISCONN	N/A	Any	None	Disconnect your terminal from your virtual machine.
DISPLAY	<G>	G		Display virtual storage on your terminal.
DMCP	<P>	C,E		Print the contents of real storage locations on a user's virtual spooled printer.
DRAIN	<S>	D		Stop spooling operations on a specified real unit record devices after the file currently being processed has been completed.
DUMP	<G>	G		Print the following on the virtual printer: Virtual PSW, general registers, floating-point registers, storage keys, and contents of specified virtual storage locations.
ECHO	<G>	G		Test terminal hardware by redisplaying data entered at the terminal.
ENABLE	<R>	A,B		Enable the previously disabled or nonenabled devices so users may access the system.
EXTERNAL	<G>	G		Simulate an external interruption for a virtual machine and return control to that machine.
FLUSH	<S>	D		Halt and immediately purge or hold the current output on a specified real unit record device.
FORCE	<O>	A		Force a logoff of any user on the system.
FREE	<S>	D		Remove a set of spool files belonging to a specified user from a system hold status.

Figure 78 (Part 3 of 8). CP Command Summary

Command	Function Type	IBM-Defined Privilege Class	User-Defined Class	Usage
HALT	<O>	A		Terminate any active channel program on a specified real device.
HOLD	<S>	D		Place user spool files in a system hold status.
INDICATE	O	A		Provides a list of statistics for all users who have the favored execution option.
	A	E		Display, at the console, the use of and contention for system processor and storage.
	G	G		Indicate resource utilization and contention.
IPL	<G>	G		Simulate IPL for a virtual machine.
LINK	<G>	G		Provide access to a specific DASD by a virtual machine.
LOADBUF	<S>	D		On a 1403 printer load the Universal Character Set(UCS) with a specified print chain/train image. On 3203, 3211, 3262, 4245, or 4248 printers, load UCS or Forms Control Buffer(FCB) with a specified image. On 3289 Model 4 printer, load the Font Offset Buffer (FOB) with the image print belt and FCB.
LOADVFCB	<G>	G		Load virtual forms control buffer for a virtual 3203, 3262, 3289E, 3211, 4245, or 4248 printer.
LOCATE	<P>	C,E		Find the addresses of CP control blocks associated with a particular user, a user's virtual device, or a real system device.
LOCK	<O>	A		Permanently locks in selected pages of real storage.
LOGOFF	N/A	Any	None	Disable access to CP.
LOGON	N/A	Any	None	Provide access to CP.
MESSAGE	N/A	Any	None	Transmit messages to other users.
	<O>	A,B		Send message text to a specified user, to primary system operator, or to one or all logged-on users.
MIGRATE	<O>	A		Activate the normal page/swap table migration routines or force a particular user's pages to the secondary device even if that user is currently active.

Figure 78 (Part 4 of 8). CP Command Summary

Command	Function Type	IBM-Defined Privilege Class	User-Defined Class	Usage
MONITOR	<O>	A,E		Initiate or override the system-generated monitor function or terminate the recording of events occurring in the real machine.
MSGNOH	<R>	B		Allow a service virtual machine to send messages to specified users without the standard header associated with the MESSAGE command.
NETWORK	O	A		Load, dump, and control operation of a 3704/3705 and control operation of a 3725 control program operating in 270x emulation mode (EP). Also control remote 3270 devices via binary synchronous lines.
	R	B		Load, dump, and control operation of a 3704/3705 and control operation of a 3725 control program operating in 270x emulation mode (EP).
NOTREADY	<G>	G		Simulate "not ready"for a device to a virtual machine.
ORDER	S	D		Place closed spool files (of a specified device type) in a different order.
	G	G		Rearrange closed spool files in a specific order.
PER	<G>	A, B, C, D, E, F, G		Monitors certain events in the user's virtual machine as they occur during program execution.
PURGE	S	D		Remove closed spool files from the system before they are printed or punched by the spooling devices or before they are read by a user.
	G	G		Remove closed spool file from the system.
QUERY	O	A		Provide status information on the real or virtual machine and miscellaneous CP functions. Also displays the status of MVS/System Extensions Support.
	R	B		Provide status information on the real or virtual machine and miscellaneous CP functions. Displays the status of the various devices.

Figure 78 (Part 5 of 8). CP Command Summary

Command	Function Type	IBM-Defined Privilege Class	User-Defined Class	Usage
	P	C		Provide system log messages and information about system users and processors.
	S	D		Provide system spooling information.
	A	E		Provide status information on the real or virtual machine and miscellaneous CP functions.
	C	F		Provide system log messages and information about system users.
	G	G		Request information about machine configuration and system status.
QVM	<O>	A		Request the transition from VM/SP environment to native mode for a particular virtual machine.
READY	<G>	G		Simulate device end interruption for a virtual device.
REPEAT	<S>	D		Increase the number of copies of an output file or place the current output file in a HOLD status increasing or not increasing the number of copies to be created.
REQUEST	<G>	G		Make an attention interruption pending for the virtual machine console.
RESET	<G>	G		Clear and reset all pending interruptions for a specified virtual device and reset all error conditions.
REWIND	<G>	G		Rewind (to load point) a tape and ready a tape unit.
SAVESYS	<A>	E		Save a virtual machine storage space with registers and PSW as they currently exist. Used in the process of creating named systems.
SCREEN	<G>	G		Allows the user to change or alter the color and extended highlighting values for his virtual machine.
SEND	<G>	G		Pass commands and message replies to disconnected virtual machine for processing.

Figure 78 (Part 6 of 8). CP Command Summary

Command	Function Type	IBM-Defined Privilege Class	User-Defined Class	Usage
SET	O	A		Establish system parameters and perform various functions to control the CP system and virtual machine options.
	R	B		Change log message, designate the unit to receive system abend dump, change time interval for a specific device class, set off monitoring for a specified class, or terminate all monitoring of missing interruptions.
	A	E		Sets paging and sets the system resource management function.
	C	F		Set the recording mode for a device and for soft errors.
	G	G		Control various functions within the virtual machine.
SHUTDOWN	<O>	A		Systematically end all virtual machine functions and checkpoint the system for an eventual warm start.
SLEEP	<G>	G		Place the virtual machine in a dormant state but allow messages to be displayed.
SMSG	<G>	G		Send special messages to specified virtual machine.
SPACE	<S>	D		Force the output on the specified printer to be single spaced for the current active spool file, regardless of the carriage control commands the actual file.
SPMODE	<O>	A		Establish or reset the single processor mode environment.
SPOOL	<G>	G		Alter spooling control options; direct a file to another virtual machine or to a remote location via the RSCS virtual machine.
SPTAPE	<S>	D		Dump spool files to tape or load spool files from tape.
START	<S>	D		Restart a spooling device after it has been drained or to change the output class that it may service.
STCP	<P>	C		Alter the contents of real storage but not real PSW or real registers.

Figure 78 (Part 7 of 8). CP Command Summary

Command	Function Type	IBM-Defined Privilege Class	User-Defined Class	Usage
STORE	<G>	G		Alter specified virtual storage locations and registers.
SYSTEM	<G>	G		Simulate RESET, CLEAR STORAGE, and RESTART buttons on a real system console.
TAG	<G>	G		Specify variable information to be associated with a spool file or output unit record device. Interrogate the current TAG text setting of a given spool file or output unit record device.
TERMINAL	<G>	G		Define or redefine the input and attention handling characteristics of your virtual console.
TRACE	<G>	G		Trace specified virtual machine activity at your terminal, spooled printer, or both.
TRANSFER	S	D		Direct one or more spool files to a specified user or reclaim reader spool files previously sent to one or more users.
	G	G		Transfer input files or reclaim input files from a specified user's virtual card reader.
UNLOCK	<O>	A		Unlock page frames previously locked by a LOCK command.
VARY	<R>	B		Mark a device available or unavailable for use by a user or the control program.
VMDUMP	<G>	G		Dump virtual machine (use VM/SP IPCS to view dump.)
WARNING	<O>	A,B		Transmit high-priority messages to a specified user or to all users.

Figure 78 (Part 8 of 8). CP Command Summary

Appendix D. Considerations for Full-Screen CMS and Windowing

The information in this section consists of a basic overview of full-screen CMS and its functions and default settings. The section also includes specific information regarding CMS, CP, System Product Editor, and application interactions with full-screen CMS. The purpose of this information is to assist users working in these areas in making a smooth transition from line-mode to full-screen CMS.

New Commands for Windowing

Several new commands have been added to VM/SP to support windowing and full-screen CMS. For the most part, these new commands are two-word commands, consisting of a verb followed by a noun. The verb is the action requested, and the noun is the object of the action.

For example, DEFINE WINDOW is the command used to create a window. The actual command name is the verb DEFINE.

Figure 79 on page 322 contains an alphabetical listing of the new full-screen CMS commands.

For further information on any command, refer to the *VM/SP CMS Command Reference.*

Command	Usage
ALARM VSCREEN	Sound the terminal alarm the next time the display is refreshed.
CLEAR VSCREEN	Erase data in the virtual screen by overwriting the data buffer with nulls.
CLEAR WINDOW	Scroll past all data in the virtual screen to which the window is connected so that no scrollable data is displayed in the window.
CURSOR VSCREEN	Position the cursor on specified line and column in a virtual screen.
DEFINE VSCREEN	Create a virtual screen.
DEFINE WINDOW	Create a window.
DELETE VSCREEN	Remove a virtual screen definition.
DELETE WINDOW	Remove a window definition.
DROP WINDOW	Move a window down in the order of displayed windows.
GET VSCREEN	Write data from a CMS file to the specified virtual screen.
HIDE WINDOW	Prevent the specified window from being displayed, and connect the window to a virtual screen.
MAXIMIZE WINDOW	Expand a window to the physical screen size.
MINIMIZE WINDOW	Reduce the size of the window to one line.
POP WINDOW	Move a window up in the order of displayed windows.
POSITION WINDOW	Change the location of a window on the physical screen.
PUT SCREEN	Make a copy of the physical screen and write the image to a CMS file.
PUT VSCREEN	Write the data from the scrollable data area of a virtual screen to a CMS file.
QUERY	The following QUERY command options have been added or changed for windowing: APL, BORDER, CHARMODE, CMSPF, CURSOR, DISPLAY, FULLREAD, FULLSCREEN, HIDE, KEY, LINEND, LOCATION, LOGFILE, NONDISP, REMOTE, RESERVED, ROUTE, SHOW, TEXT, VSCREEN, WINDOW, and WMPF. For specific information regarding the usage of each command, refer to the *VM/SP CMS Command Reference.*
REFRESH	Update virtual screens and their associated windows and refresh the screen.
RESTORE WINDOW	Return a maximized or minimized window to its size and location prior to the maximize or minimize.
ROUTE	Direct data of a particular message class to a virtual screen.
SCROLL	Move a window to a new location on the virtual screen.

Figure 79 (Part 1 of 2). CMS Command Summary

Command	Usage
SET	The following SET command options have been added or changed for windowing: APL, BORDER, CHARMODE, CMSPF, FULLREAD, FULLSCREEN, LINEND, LOCATION, LOGFILE, NONDISP, REMOTE, RESERVED, TEXT, VSCREEN, WINDOW, and WMPF. For specific information regarding the usage of each command, refer to the *VM/SP CMS Command Reference*.
SHOW WINDOW	Place a window on top of all other displayed windows and connect the window to a virtual screen.
SIZE WINDOW	Change the number of lines and columns for a specified window.
WAITREAD VSCREEN	Use from an EXEC to update the virtual screen with data, refresh the physical screen, and wait for the next attention interrupt.
WAITT VSCREEN	Update the virtual screen with data.
WRITE VSCREEN	Enter information in a virtual screen.

Figure 79 (Part 2 of 2). CMS Command Summary

Several single-character Border commands have also been added for windowing and full-screen CMS. Border commands are windowing commands that you may enter in the corners of a window border. These commands are summarized below:

B	Scrolls the window backward
C	Clears the window of scrollable data
D	Drops the window
F	Scrolls the window forward
H	Hides the window
L	Scrolls the window to the left
M	Changes the location of the window
N	Minimizes the window
O	Restores the window
P	Pops the window
R	Scrolls the window to the right
S	Changes the size of the window
X	Maximizes the window

For further information on these Border commands, refer to the *VM/SP CMS Command Reference.*

General Information on Full-Screen CMS

There are many advantages to using full-screen CMS. When you set full-screen CMS on, you can type commands from almost anywhere on the physical screen. You can scroll forward and backward through your CMS session to see commands you entered previously and CMS responses to these commands. To re-issue any command, you do not need to re-type the entire command. Instead, you can scroll back to where the command was previously entered, retype any letter, and press ENTER to re-issue the command.

During your full-screen CMS session, messages and other output appear in windows on your physical screen and can be viewed without leaving your current work environment.

Virtual Screens and Windows

In full-screen CMS, a virtual screen (vscreen for short) is a presentation space where data can be maintained. A window is an area on the physical screen that can be used to display and manipulate virtual screen data. For further details on windows and vscreens, refer to Chapter 9, "Looking at VM/SP Through Windows."

Following is a listing of the default virtual screens and windows which are available to you in full-screen CMS.

Virtual Screen	Lines	Cols	Rtop	Rbot	Dcolor	Options
STATUS	1	Pscr	0	0	White	PROTECT
NETWORK	16	70	2	0	Blue	PROTECT
WARNING	4	70	2	0	Red	PROTECT
MESSAGE	20	70	2	0	White	PROTECT
CMS	120	Pscr	2	5	Green	NOPROTECT

Figure 80. Default Virtual Screens

Window	Lines	Cols	Psline	Pscol	Options
STATUS	1	Pscr	-1	1	FIXED NOBORDER NOPOP NOTOP

Figure 81 (Part 1 of 2). Default Windows

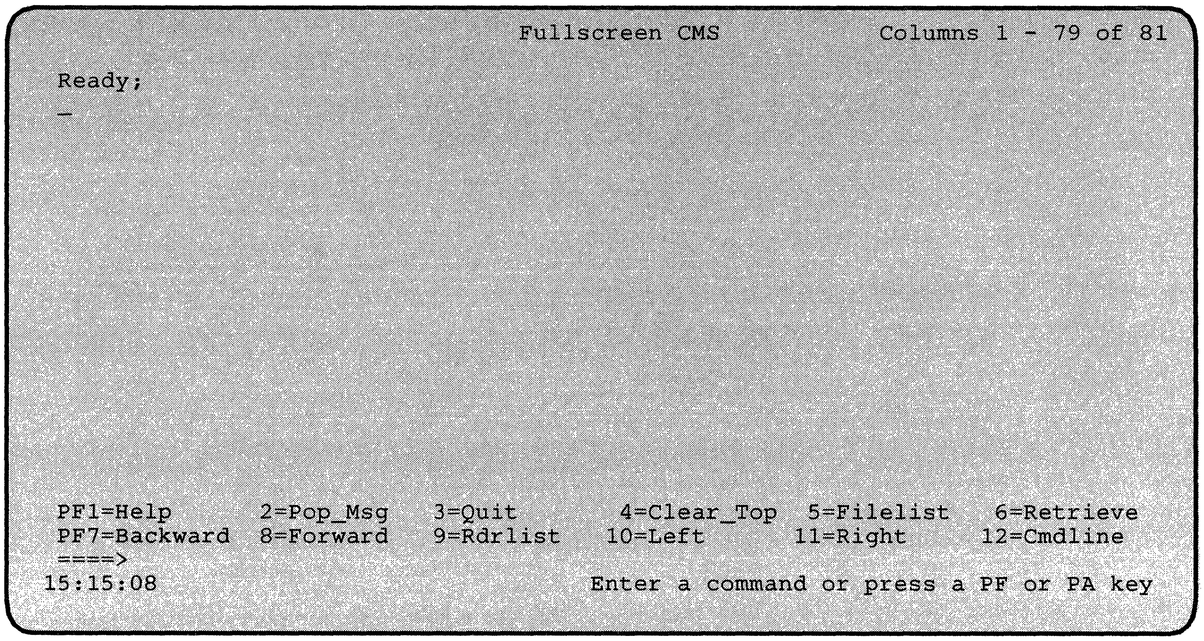
Window	Lines	Cols	Psline	Pscol	Options
CMS	Pscr	Pscr	1	1	FIXED BORDER NOPOP TOP
NETWORK	8 (max.)	71	-12	7	VARIABLE BORDER NOPOP TOP
WARNING	6 (max.)	71	3	3	VARIABLE BORDER POP TOP
MESSAGE	8 (max.)	71	11	3	VARIABLE BORDER POP TOP
CMSOUT	8	75	9	3	VARIABLE BORDER POP TOP

Figure 81 (Part 2 of 2). Default Windows

Definitions of the terminology and abbreviations used in these tables are located at the end of Chapter 15, "Customizing Full-Screen CMS."

Screen Organization

Once you enter the command SET FULLSCREEN ON, your screen will be organized differently from the way it appears in line-mode CMS. Following is an example of the screen layout for full-screen CMS.



```
                                   Fullscreen CMS          Columns 1 - 79 of 81
  Ready;
  _










  PF1=Help      2=Pop_Msg   3=Quit      4=Clear_Top  5=Filelist   6=Retrieve
  PF7=Backward  8=Forward   9=Rdrlist   10=Left      11=Right     12=Cmdline
  ====>
 15:15:08                                Enter a command or press a PF or PA key
```

Figure 82. Screen Organization

Chapter 9, "Looking at VM/SP Through Windows" contains a detailed description of each of the areas of the screen.

You will also notice that the line-mode VM status notices will be replaced by the following full-screen status notices:

- Executing a command
- Enter your response in vscreen *'vname'*
- Scroll forward for more information in vscreen *vname'*
- Enter a command or press a PF or PA key

These status notices are also fully explained in Chapter 9, "Looking at VM/SP Through Windows."

Default Settings

In full-screen CMS, you are provided with a set of PF keys called CMSPF keys. These keys are set by default to perform certain windowing functions as listed in the table below.

CMSPF Key	Pseudonym	Command
CMSPF 1	Help	ECHO HELP
CMSPF 2	Pop_Msg	NOECHO #WM POP WINDOW MESSAGE *
CMSPF 3	Quit	NOECHO SET FULLSCREEN SUSPEND
CMSPF 4	Clear_Top	NOECHO #WM CLEAR WINDOW =
CMSPF 5	Filelist	ECHO EXEC FILELIST
CMSPF 6	Retrieve	RETRIEVE
CMSPF 7	Backward	NOECHO #WM SCROLL BACKWARD CMS 1
CMSPF 8	Forward	NOECHO #WM SCROLL FORWARD CMS 1
CMSPF 9	Rdrlist	ECHO EXEC RDRLIST
CMSPF 10	Left	NOECHO #WM SCROLL LEFT CMS 10
CMSPF 11	Right	NOECHO #WM SCROLL RIGHT CMS 10
CMSPF 12	Cmdline	NOECHO CURSOR VSCREEN CMS -2 8 (RESERVED

Figure 83. CMSPF Key Settings

You can change any of these settings by using the command SET CMSPF, followed by the PF key number (represented as nn).

You can set your CMSPF keys to perform any command, including the windowing commands listed under "The WM Environment" on page 328, with the exception of HELP. (HELP can be issued from the WM window but cannot be issued as a #WM command).

If you wish to set your CMSPF keys to perform any of these windowing commands, you can begin the command with "#WM" to indicate that the windowing command will be executed immediately in the CMS window. While you can set your CMSPF keys to these windowing commands without using "#WM," if you do so, the PF key commands may be executed after other commands which are pending at the time you press the PF key.

For further information on the default settings and how to change them, refer to Chapter 9, "Looking at VM/SP Through Windows."

The PA1, PA2, and CLEAR keys also have special meanings in full-screen CMS. The PA1 key pops the WM window. The PA2 key and the CLEAR key serve the same purpose and can be used to scroll the top window forward. Refer to Chapter 9, "Looking at VM/SP Through Windows" for details.

Full-screen CMS also provides you with several commands to control how you receive messages from the system, other users, and your programs. These commands are discussed in Chapter 8, "Communicating with Other

Computer Users" and are fully described in the *VM/SP CMS Command Reference.*

You may find it useful to refer to the following table which contains information regarding the default settings for message routing. The terms used in the table are discussed at the end of Chapter 15, "Customizing Full-Screen CMS."

Message Class	Virtual Screen	Options
CMS	CMS	NOALARM NONOTIFY
CP	CMS	NOALARM NONOTIFY
MESSAGE	MESSAGE	ALARM NOTIFY
WARNING	WARNING	ALARM NOTIFY
SCIF	MESSAGE	NOALARM NONOTIFY
NETWORK	NETWORK	NOALARM NOTIFY

Figure 84. Default Settings for Message Routing

The WM Environment

The WM window is a special window available to you for window manipulation. That is, it can be used to drop, move, or change the size of other windows. Depending on how the WM window was invoked, one of the following messages will appear in the window:

- Active window overlaid; enter a windowing command or press a PF key
- Output displayed; enter a windowing command or press a PF key
- Enter a windowing command or press a PF key

Chapter 9, "Looking at VM/SP Through Windows" provides further information on when each message is displayed.

The WM window will automatically be displayed on your screen in the following special situations:

- When your entire screen is protected and the active window is overlaid. (For example, this situation could occur if you maximize a window which is protected so that it fills the entire screen and covers all other windows). Chapter 9, "Looking at VM/SP Through Windows" gives a detailed explanation of the WM window and provides an example of this type of scenario.

- When you run an application that uses the CONSOLE macro to perform I/O and
  - The CMS virtual screen is updated, or
  - Any virtual screen (other that CMS) is updated and a pop-type window is showing it.

  In this instance, you can simply drop the WM window to return to the application.

If you wish, you can also pop the WM window at any time during your full-screen CMS session. To do this, you would simply enter the command POP WINDOW WM or press the PA1 key (which in full-screen CMS defaults to the command POP WINDOW WM).

If you pop the WM window, or when it appears on your screen, you will have access to a special set of PF keys called the WMPF keys, which you can use to manipulate other windows on your screen. The default setting for these keys are listed below.

WMPF Key	Pseudonym	Command
WMPF 1	Help	NOECHO HELP
WMPF 2	Top	NOECHO SCROLL TOP =
WMPF 3	Quit	NOECHO DROP WINDOW WM
WMPF 4	Clear	NOECHO CLEAR WINDOW =
WMPF 5	Copy	NOECHO PUT SCREEN COPY SCREEN
WMPF 6	Retrieve	RETRIEVE
WMPF 7	Backward	NOECHO SCROLL BACKWARD = 1
WMPF 8	Forward	NOECHO SCROLL FORWARD = 1
WMPF 9	Maximize	NOECHO MAXIMIZE WINDOW =
WMPF 10	Left	NOECHO SCROLL LEFT = 10
WMPF 11	Right	NOECHO SCROLL RIGHT = 10
WMPF 12	Restore	NOECHO RESTORE WINDOW =

Figure 85. WMPF Key Settings

In addition, you can manipulate other windows on your screen by entering windowing commands in the WM window. Any of the windowing commands listed below can be entered from the WM window:

CLEAR WINDOW	HELP	MINIMIZE WINDOW
CP	HIDE WINDOW	POP WINDOW
DROP WINDOW	MAXIMIZE WINDOW	POSITION WINDOW
PUT SCREEN	QUERY WINDOW	SET RESERVED
QUERY BORDER	QUERY WMPF	SET WINDOW
QUERY HIDE	RESTORE WINDOW	SET WMPF
QUERY LOCATION	SCROLL	SHOW WINDOW
QUERY RESERVED	SET BORDER	SIZE WINDOW
QUERY SHOW	SET LOCATION	

Specific Migration Considerations

The remainder of this section provides specific information for users of CP, CMS, XEDIT, and applications. It is recommended that you review these items carefully so that you will understand the benefits of full-screen CMS as well as any adjustments you may need to make to utilize full-screen CMS capabilities fully.

CMS Considerations

- CMS Immediate commands (including #WM and #CP in full-screen CMS), Border commands, and windowing commands entered in the WM window do not support synonyms and cannot be translated to another language.

- When full-screen CMS is on, your PF keys are called CMSPF keys and are set by default to perform windowing functions such as scrolling the CMS window, popping the MESSAGE window, and clearing the window on top. To override these settings or to make your CMSPF keys equivalent to your line-mode CP PF keys, set full-screen CMS on and then use the SET CMSPF command. Changing your CMSPF keys does not affect your CP PF keys.

- Fullscreen CMS sets CMSPF 6 to RETRIEVE. You do not have to define your own key to perform this function unless you want to override the default. Another way to retrieve commands that you previously entered is to scroll your CMS window back so that the command you wish to re-enter is visible. Then, position the cursor over the command, re-type any character, and press ENTER. The command is re-issued. (The command echo and output are appended to the bottom of the CMS virtual screen by default).

- In full-screen CMS, virtual screen output may be logged to CMS files. Messages and warnings are logged by default. For more information, refer to the SET LOGFILE and SET FULLSCREEN ON commands in the *VM/SP CMS Command Reference.* The CP SPOOL CONSOLE START command does not log full-screen CMS interactions.

- To copy a full-screen CMS screen image, use the PUT SCREEN command. This command copies the image of your physical screen to a CMS file which you can then XEDIT or print.

- To tailor window and virtual screen attributes and extended attributes, use the DEFINE VSCREEN, SET VSCREEN, WRITE VSCREEN, and SET BORDER commands. The CP SCREEN command has no effect on the attributes of full-screen CMS output.

- To interrupt an EXEC that is reading data from your terminal, type an Immediate command prefixed by "#" (or the current LINEND character). For example, you would type "#HT" to halt typing or "#HI" to halt interpretation.

- In full-screen CMS, the BLIP character is not displayed immediately. When execution of the command is completed, the screen is refreshed and any blip characters that have been queued are displayed.

- If you wish to type ahead the next command (or commands) while a command is executing, do so on the full-screen CMS command line. Typing ahead in the input/output area in the CMS window is not recommended. When the currently-executing command completes, the screen is refreshed with new output written to the CMS window. This new output may overlay the command that you were in the process of typing.

- Use the command SET VSCREEN CMS NOTYPE to suppress output to the CMS virtual screen. The CP SPOOL CONSOLE NOTERM command does not suppress full-screen CMS output.

- To enter long commands, type in the full-screen CMS input/output area. If you wish to enter long commands from the command line, drop or hide the STATUS window. (See the DROP WINDOW and HIDE WINDOW commands in the *VM/SP CMS Command Reference* for details). CMS command input is limited to 255 characters.

- When you press a PF or PA key, any input on the screen that has not been processed is discarded except input that is typed on the command line. If the key that was pressed does not update the command line, then input on the command line is rewritten.

- PF keys set to execute Immediate commands (the key definition is preceded by #WM or #CP) are executed immediately. All other key definitions are stacked.

- On terminals with 80 columns, command output that is 80 characters wide cannot be viewed completely without scrolling the CMS window to the right. This is because each line of CMS output is typically preceded by two Start Fields, one in the CMS vscreen and one that is placed at the start of each line when the physical screen is displayed. (See the WAITREAD VSCREEN command in the *VM/SP CMS Command Reference* for further information). Therefore, the CMS window, in its default position at column one of the CMS vscreen, can display columns 1 through 79 of the output. The remaining output can be viewed by scrolling to the right.

- If the virtual screen is wider than the window displaying it, then a field which wraps onto multiple lines in the virtual screen cannot be displayed as a single field on the screen. Certain keys on the terminal

(ERASE EOF, DELETE, and INSERT MODE) will only work on fields on the screen and not on the entire field in the virtual screen. In this case, the field is reconstructed using the changes on the screen and the parts of the field that are in the virtual screen (including parts which are not displayed on the screen). Note that the function of these keys will always work on fields on the screen; whenever a window vertically splits another window, these keys will not work on the entire field in the virtual screen.

- If you disconnect from full-screen CMS, when you reconnect, you may get a "MORE..." status. Press the CLEAR key to return to full-screen CMS.

- The screen is refreshed when there is a large amount of output waiting to be displayed. When a program or command issues a number of lines of output which is equal to the number of lines on the physical screen less one, the screen is refreshed.

- In full-screen CMS, there is no terminal escape character (ESCAPE), line delete character (LINEDEL), or character delete character (CHARDEL).

- Because CMS is a single-tasking system, CMS windowing only supports one active virtual screen on the physical screen. Other interactive virtual screens may be displayed on the physical screen but are protected.

- The IUCV Message All System Service can stack up to 255 messages at any one time. If this limit is exceeded, any additional incoming messages are sent directly to the terminal.

- If the window, virtual screen, and physical screen do not have the same number of columns, it is recommended that you define the window with one column greater than the number of columns in the vscreen that it is displaying. This provides for the additional field definition character (Start Field) which is necessary for the proper display of the window on the screen and ensures that a maximum number of columns of virtual screen data are displayed.

CP Considerations

- In order to enter CP commands while in full-screen CMS, preface the commands with "CP" or "#CP." The output from the CP commands will be displayed by full-screen CMS. If you wish to drop into CP during your full-screen CMS session, type "CP" and press ENTER. This will result in a cleared screen with a "CP READ" status. While you can then enter CP commands and receive output to your terminal, any CP commands you enter while in the CP READ status will not be displayed in full-screen CMS.

- "#CP" (the default LINEND character, followed by "CP") is processed by full-screen CMS before it is processed by CP. In full-screen CMS, "#CP" is treated as a CMS Immediate command.

- If the CP SLEEP command is issued while full-screen CMS is on, it may appear that your terminal is hung. Any write to the terminal will unlock the keyboard. Refer to the CP SLEEP command in the *VM/SP CP Command Reference* for further details.

- The CP SCREEN command has no effect on the attributes of full-screen CMS output. To tailor window and virtual screen attributes and extended attributes, use the DEFINE VSCREEN, SET VSCREEN, WRITE VSCREEN, and SET BORDER commands.

- The CP SPOOL CONSOLE NOTERM command does not suppress full-screen CMS output. Use the command SET VSCREEN CMS NOTYPE to suppress output to the CMS virtual screen.

- All messages classified as message class MESSAGE are displayed with headers. (See the ROUTE command in the *VM/SP CMS Command Reference* for further information). This includes messages sent via the CP MESSAGE and the CP MSGNOH commands.

- CP does not edit lines entered in a full-screen environment. In full-screen CMS, there is no terminal escape character (ESCAPE), line delete character (LINEDEL), or character delete character (CHARDEL).

System Product Editor Considerations

- XEDIT no longer carries out its own I/O. Windowing functions are responsible for XEDIT I/O. As a result, certain CMS settings affect the XEDIT environment, especially the following:

  - SET LANGUAGE (affects Double-Byte Character Sets (DBCS) display and the nondisplayable character set)
  - SET APL/TEXT
  - SET FULLREAD
  - SET NONDISP
  - SET REMOTE

- The XEDIT SET BRKKEY works differently. XEDIT no longer restores the break key to whatever it was when SET BRKKEY ON was issued. Instead, if BRKKEY was set in XEDIT, the CP setting remains when you are no longer in XEDIT.

  In addition, the initial SET BRKKEY setting now reflects the CP TERMINAL BRKKEY setting. It is no longer always ON by default.

- The default PA1 key for XEDIT (and the NAMES and SENDFILE commands) is now COMMAND CMS POP WINDOW WM if BRKKEY is not assigned to the PA1 key.

- QUERY and EXTRACT return virtual screen information rather than physical screen information.

- COPYKEY copies the content of the virtual screen, rather than the content of the physical screen, into the printer spool.

- The initial SET ETMODE setting is no longer OFF by default. This setting is now based on whether the terminal in use is capable of handling double-byte characters.

- If you are using a 3277 terminal and you issue QUERY PF, you now get the settings for 24 PF keys instead of just 12.

- Nullkey is an existing option you can specify on any XEDIT PF or PA key or on the ENTER key. Now, the nullkey function replaces trailing blanks with nulls on the field of the screen that contains the cursor. If the cursor is on a prefix area, the nulls are written to the field of the file line associated with that prefix area. Before, the nullkey function wrote the nulls on the line containing the cursor.

- Although you can define your own XEDIT virtual screen and assign to it any valid default attributes and extended attributes, XEDIT overrides these when it writes fields. You should continue to use the XEDIT SET COLOR subcommand to change the XEDIT screen attributes.

Considerations for Writing Applications

- If the CP SLEEP command is issued while full-screen CMS is on, it may appear that your terminal is hung. Any write to the terminal will unlock the keyboard. Refer to the CP SLEEP command in the *VM/SP CP Command Reference* for further details.

- Some programs imbed the hexadecimal code "1D" to affect the highlighting and color attributes of output. In full-screen CMS, X'1D' is a nondisplayable character and does not affect the attributes of data following it. The DEFINE VSCREEN, SET VSCREEN, and WRITE VSCREEN commands (as well as the LINEWRT macro) allow programs to specify attributes for data in full-screen CMS.

- Error messages generated from windowing commands are displayed based on how the command was invoked:

  - When called from a program or from an EXEC procedure with &CONTROL NOMSG, then no message is displayed and only the return code is set.

  - When called from an EXEC procedure with ADDRESS COMMAND, then the variable *message.1* is set to the message text and *message.0* is set to 1 to indicate the number of message variables set.

  - In other cases, such as a command entered from the command line or from an EXEC procedure with ADDRESS CMS, the message is displayed at the terminal. These messages are edited based on the CP EMSG setting.

- When returning to full-screen CMS from an application that performs its own full-screen management, your screen may contain mixed data.

Press the CLEAR key to scroll forward and refresh the screen. Alternatively, you can enter the command SET FULLSCREEN SUSPEND before invoking the application.

- The following messages are not trapped by the IUCV Message All System Service and are sent directly to the terminal:
  - Asynchronous CPCONIO (including PER/TRACE events)
  - EMSGs not generated as part of a DIAG X'08' instruction
  - Accounting messages.
- Certain applications must be changed to work correctly in the full-screen CMS environment:
  - Programs which issue a 3215 SIO to do a line-mode read
  - Programs which issue DIAG X'58' to do full-screen I/O

  Line-mode output and prompts written by these applications will not be displayed immediately in the full-screen CMS environment. Also, messages and warnings from other computer users will not be displayed until the user exits these applications. These applications should be changed to use new CMS macros:

  - Change 3215 SIO to use LINERD macro
  - Change DIAG X'58' to use CONSOLE macro

  Alternatively, to avoid problems viewing output during the execution of such programs, you can temporarily suspend full-screen CMS (SET FULLSCREEN SUSPEND) before running the application and then resume your session (SET FULLSCREEN RESUME) upon exiting the application.
- When full-screen CMS is on, most CMS console output is not passed to CP. In addition, applications that use the IUCV Message System Service (*MSG) and SET VMCONIO IUCV will not trap all CMS output. Prior to running such applications, it is recommended to suspend full-screen CMS.
- Full-screen CMS initialization issues the CP TERMINAL BRKKEY NONE command. Application developers may want to reset the CP TERMINAL BRKKEY to PA1 when debugging.
- You can specify the EXIT parameter for the OPEN function of the CONSOLE macro instruction to handle unrequested device interrupts.
- If EXIT is specified, **do not** define an interruption routine via the HNDINT macro for the same device. Use of the CONSOLE and HNDINT macros is mutually exclusive. CONSOLE OPEN with EXIT supercedes an HNDINT routine when the interrupt is requested. Therefore, if you want to do I/O to a 3270 device, use the CONSOLE macro instead of the HNDINT macro.

Summary of Changes

Summary of Changes
for SC19-6210-4
for VM/SP Release 5

This publication has been restructured for Release 5 to meet the needs of general users. Much of the specific programming information has been relocated to the system programmer's reference books in order to better organize the VM/SP library as a whole. The following chart shows which chapters have been relocated and the destination books:

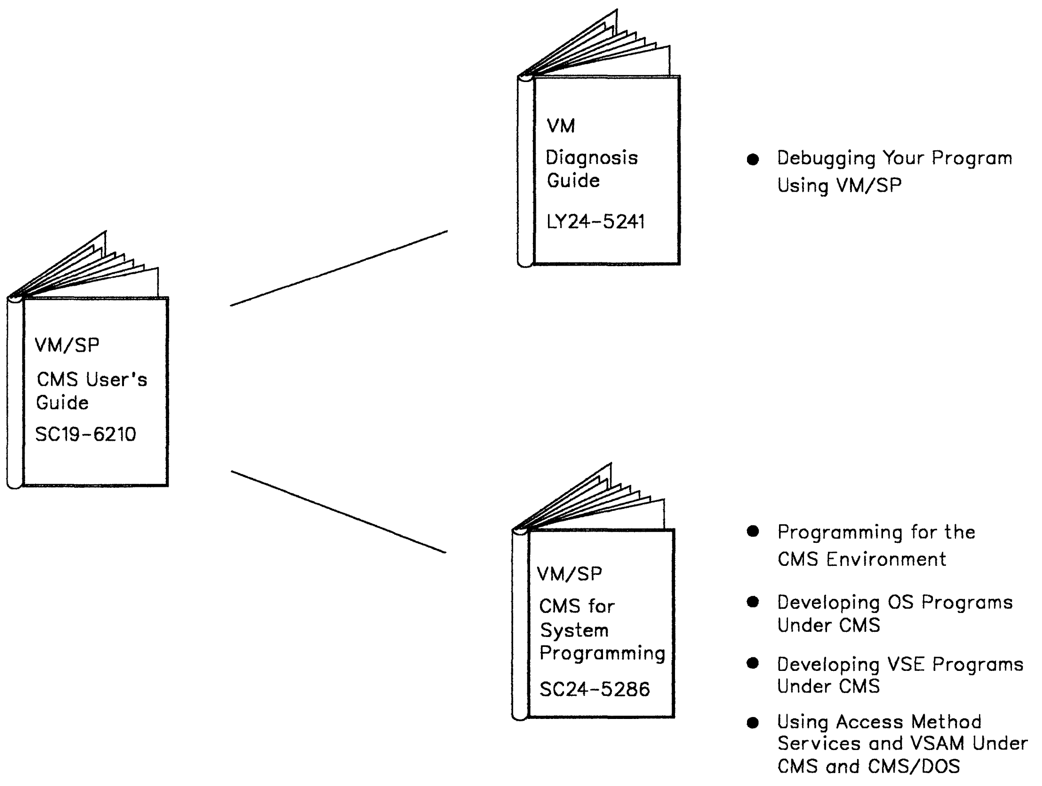

VM Diagnosis Guide LY24-5241
• Debugging Your Program Using VM/SP
VM/SP CMS User's Guide SC19-6210
VM/SP CMS for System Programming SC24-5286
• Programming for the CMS Environment
• Developing OS Programs Under CMS
• Developing VSE Programs Under CMS
• Using Access Method Services and VSAM Under CMS and CMS/DOS

Figure 86. Restructuring of the CMS User's Guide

The remaining information within this publication has been revised as follows:

- "Part 1: Getting Acquainted with VM/SP" has been retitled and now includes "Chapter 5, Using the HELP Facility" (previously Chapter 19 in Part 4).
- "Part 2: Working with VM/SP" has been retitled and now includes "Chapter 6, Editing Your Files" (previously Chapter 5 in Part 1), "Chapter 7, Using Real Printers, Punches, Readers, and Tapes" (previously Chapter 6 in Part 1), "Chapter 8, Communicating with Other Computer Users" (previously Chapter

7 in Part 1), "Chapter 9, Looking at VM/SP Through Windows" (new for this release), and "Chapter 10, Using the CMS Batch Facility" (previously Chapter 12 in Part 2).

- In "Part 3: Learning to Use EXECs," the chapter entitled "Exchanging Data Between Programs Through the Stack" (previously Chapter 17) has been deleted.

- "Part 4: Tailoring Your System" has been retitled and now includes "Chapter 15, Customizing Full-Screen CMS" and "Chapter 16, Tailoring the HELP Facility."

How to Obtain Prior Editions of This Publication

To obtain editions of this publication that pertain to earlier releases of VM/SP, you must use the pseudo-number assigned to the respective edition when ordering. For:

Release 4, order ST00-1584

Release 3, order ST00-1358

Release 2, order SQ19-6210

Release 1, order ST19-6210.

New Commands for Release 5 of VM/SP

The following CMS commands are new for this release:

ALARM VSCREEN	Sound the terminal alarm when the display is refreshed
CLEAR VSCREEN	Erase data in the virtual screen
CLEAR WINDOW	Scroll past all data in the virtual screen
CONVERT COMMANDS	Convert a CMS file containing Definition Language for Command Syntax (DLCS) statements into an internal form for the parsing facility
CMSSERV	Start IBM Cooperative Processing Communications between your VM/SP host system and your work station (IBM Personal Computer)
CURSOR VSCREEN	Position the cursor on a specified line and column in a virtual screen
DEFINE VSCREEN	Create a virtual screen
DEFINE WINDOW	Create a window
DELETE VSCREEN	Remove a virtual screen definition
DELETE WINDOW	Remove a window definition
DROP WINDOW	Move a window down in the order of displayed windows

GENMSG	Convert a message repository into an internal form
GET VSCREEN	Write data from a CMS file to the specified virtual screen
HIDE WINDOW	Prevent the specified window from being displayed, and connect the window to a virtual screen
LANGGEN	Save all text files for a language in DCSS, and/or save CP message repository
LANGMERG	Combine all language-related files for an application into one text file
MAXIMIZE WINDOW	Expand a window to the physical screen size
MINIMIZE WINDOW	Reduce the size of the window to one line
MOREHELP	Obtain either additional or related information about the latest valid HELP command you issued
PARSECMD	Call the parsing facility from within an EXEC
POP WINDOW	Move a window up in the order of display windows
POSITION WINDOW	Change the location of a window on the physical screen
PUT SCREEN	Make a copy of the physical screen and write the image to a CMS file
PUT VSCREEN	Write the data from the scrollable data area of a virtual screen to a CMS file
QUERY	The following QUERY command options have been added or changed for Release 5: APL, BORDER, CHARMODE, CMSPF, CURSOR, DISPLAY, FULLREAD, FULLSCREEN, HIDE, INSTSEG, KEY, LINEND, LOCATION, LOGFILE, NONDISP, REMOTE, RESERVED, ROUTE, SHOW, TEXT, TRANSLATE, VSCREEN, WINDOW, and WMPF
REFRESH	Update virtual screens and their associated windows and refresh the screen
RESTORE WINDOW	Return a maximized or minimized window to its original size and location
ROUTE	Direct data of a particular message class to a virtual screen
SCROLL	Move a window to a new location on the virtual screen

SET	The following SET command options have been added or changed for Release 5: ABBREV, APL, BORDER, CHARMODE, CMSPF, FULLREAD, FULLSCREEN, INSTSEG, LANGUAGE, LINEND, LOCATION, LOGFILE, NONDISP, REMOTE, RESERVED, TEXT, TRANSLATE, VSCREEN, WINDOW, and WMPF
SHOW WINDOW	Place a window on top of all other displayed windows and connect the window to a virtual screen
SIZE WINDOW	Change the number of lines and columns for a specified window
WAITREAD VSCREEN	Use from an EXEC to update the virtual screen with data, refresh the physical screen, and wait for the next attention interrupt
WAITT VSCREEN	Update the virtual screen with data
WRITE VSCREEN	Enter information in a virtual screen
XMITMSG	Retrieve a message from a CMS message repository file or your own message repository file

In addition, several single-character Border commands have been added to manipulate windows in the full-screen CMS environment:

B	Scrolls the window backward
C	Clears the window of scrollable data
D	Drops the window
F	Scrolls the window forward
H	Hides the window
L	Scrolls the window to the left
M	Changes the location of the window
N	Minimizes the window
O	Restores the window
P	Pops the window
R	Scrolls the window to the right
S	Changes the size of the window
X	Maximizes the window

Integration of Functional Enhancements to Release 4

VM/SP now supports the IBM 3380 Direct Access Storage Device, Models AE4/BE4.

National Language Support

National Language Support (NLS) has been added to VM/SP to provide support for languages other than American English:

- SET and QUERY commands determine the languages supported and set your virtual machine to operate within a particular language environment
- The Central Message Facility, a consolidated file of all system messages and responses, is used to simplify the translation process
- The parser facility parses and translates command name arguments, eliminating the need for syntax checking in programs and allowing users to enter programs in their own language by modifying the Definition Language for Command Syntax (DLCS).

Enhanced 3270 Support

The following changes have been made to VM/SP to provide enhanced support for 3270-type terminals:

- A protected application environment protects interactive application users from accidentally entering the CP environment
- The VM logo message at the top of the screen has been changed from "VM/370" to "VIRTUAL MACHINE/SYSTEM PRODUCT"
- Users can now log on directly from the logo screen
- A new prompting screen and error message (DMKCFM288E LOGON FROM THE INITIAL SCREEN WAS UNSUCCESSFUL) have been added.

EXECs and Macros in a Discontinguous Shared Segment

An optional facility has been added to VM/SP to allow installations to build an Installation Discontiguous Shared Segment (DCSS) to contain frequently used EXECs and Editor Macros. Users can access the DCSS and share the same executing copy of the EXECs.

CMS Session Services

VM/SP now provides the end-user with window and virtual screen functions. When the command SET FULLSCREEN ON is issued, CMS is in a window. This allows the user to enter commands from almost anywhere on the physical screen, scroll through data, and log data into files. The user can also view messages and other information through windows on the physical screen.

System Profile

SYSPROF EXEC has been added to VM/SP to contain some of the CMS initialization functions, such as setting up the default CMS environment, which were previously done in a module. This allows the system administrator to tailor the CMS environment for the users simply by changing the EXEC.

Advanced Printer Subsystem

Advanced Printer Subsystem (APSS) has been added to VM/SP to provide a destination option for spool files. This allows users to select a specific printer or punch to process print, punch, or console files.

HELP Facility

The VM/SP HELP Facility has been enhanced to improve performance and usability and to include National Language Support:

- HELP screens and syntax notation have been simplified
- A BRIEF layer of HELP has been added to provide information on frequently-used commands
- New HELP command options (BRIEF, DETAIL, RELATED, EXTEND, TYPE, and NOTYPE) have been added
- The MOREHELP command has been added to provide additional detail or related HELP information
- New keywords have been added for specifying control sections in HELP files
- The ability to toggle between the BRIEF, ALL, and RELATED layers of HELP has been added.

Miscellaneous

- This major revision incorporates minor technical and editorial changes.

Summary of Changes for SC19-6210-3 for VM/SP Release 4

How to Obtain Prior Editions of This Publication

To obtain editions of this publication that pertain to earlier releases of VM/SP, you must use the pseudo-number assigned to the respective edition when ordering. For:

Release 3, order ST00-1358

Release 2, order SQ19-6210

Release 1, order ST19-6210.

New Commands for Release 4 of VM/SP

The following CMS commands are new for this release:

EXECDROP- Purges storage resident EXECs.

EXECLOAD - Loads EXECs into storage.

EXECMAP - Provides a list of storage resident EXECs.

EXECSTAT - Provides the status of a specified EXEC.

HELPCONV - Converts a specified file into a formatted HELP file, leaving the .CS, .CM, and .MT control words in the file.

MACLIST - Displays a list of all members in a specified MACLIB, with the ability to edit and issue commands from the list.

Integration of Functional Enhancements to Release 3

Document support of the following:

- IBM 3800 Printing Subsystems, Models 1 and 3
- IBM 3370 Direct Access Storage Device, Models A2 and B2
- 3290 Information Panel
- IBM 4248 Printer.

CMS Macro Library Enhancements

Document the changes to CMS when you are using CMS macro libraries:

- The new MACLIST command displays a list of all members in a specified macro library, with the ability to edit and issue commands from the list. You can use the DEFAULTS command to customize the options for the MACLIST command.
- The new XEDIT MEMBER option allows you to specify the name of a member to be created and/or edited in a specified macro library.
- Individual member names can now be specified with the MAP operand of the MACLIB command. MACLIB MAP output can be stacked using the new STACK, LIFO, and FIFO options.

XEDIT Mixed Case Messages

Document changes to XEDIT screens for messages that appear in mixed case.

Logon Procedure

Document the logon instructions that are displayed after you clear the logo.

DDR Command Enhancement

Document support of the DDR command COMPACT option used for compacting tape output.

HELP Facility

Document the changes to the HELP Facility:

- The addition of HELP Task Menus
- The addition of the IPCS component
- The new HELPCONV command used to convert a file into a formatted HELP file
- The new operands and options of the HELP command
- The addition of the .MT (MENU TYPE) HELPCONV format word.

EXECIO command VAR and STEM options

The STEM and VAR options allow you to use the EXECIO command directly with REXX or EXEC 2 variables.

EXECs in Storage

EXECs that you want to remain storage resident can be loaded into storage with the new EXECLOAD command. The new EXECDROP command will drop a storage resident EXEC from storage. Information about storage resident EXECs can be displayed or stacked with the new EXECMAP command and the status of a specific EXEC can be determined by using the EXECSTAT command.

CMS Command Search Order

Document the search for storage resident EXECs in the CMS Command Search Order description.

Migration of CMS Commands into the Nucleus

The following commands are now nucleus resident commands:

- ACCESS
- DLBL
- FILEDEF
- RELEASE
- SET

Tape Support

Document the changes in tape support:

- Support of the IBM 3480 Magnetic Tape Subsystem
- New TAPE command options, TRANSFER BUFFERED and TRANSFER IMMEDIATE, for writing tapes to the 3480 Magnetic Tape Subsystem
- Support of 38K density for 18-track tapes
- Support of multi-volume files for tapes with standard labels used in OS simulation.

New VM/SP Component

Document support of the Interactive Problem Control System (IPCS) component of VM/SP.

Miscellaneous

- This major revision incorporates minor technical and editorial changes.

Summary of Changes
for SC19-6210-2
for VM/SP Release 3

Reorganization of this Manual

- "Part 1. Understanding CMS" contains a new chapter about communicating with other computer users. Documentation on the CMS editor has been moved to "Appendix A." "Chapter 4. What You Can Do with CMS Commands" provides an overview of the operations that you might need to perform and the commands that you can use to perform these operations.

- The chapters in "Part 2. Program Development Using CMS" appear in a new order within the part.

- "Part 3: Learning to Use EXECs" contains overviews of the three EXEC interpreters. Documentation concerning the CMS EXEC Facility is now in "Appendix B."

- The chapters in "Part 4. The HELP Facility" have been condensed to two chapters.

New Commands for Release 3 of VM/SP

The following CMS commands are new for this release:

CATCHECK - Allows CMS VSAM users to invoke the VSE/VSAM Catalog Check Service Aid to verify a complete catalog structure.

EXECOS - Resets the OS and VSAM environments under CMS without returning to the interactive environment.

EXECUPDT - Used to apply updates to a System Product Interpreter source program and create an executable version of the program.

IMMCMD - Establishes or cancels Immediate commands from within an EXEC.

RESERVE - Allocates all available blocks of a 512-, 1K-, 2K-, or 4K-byte block formatted minidisk to a unique CMS file.

New Immediate Commands

HI - Halt interpretation terminates execution of all currently executing System Product Interpreter or EXEC 2 EXECs without destroying the environment as HX would.

TE - Trace end stops all tracing of your System Product Interpreter or EXEC 2 programs or macros.

TS - Trace Start starts tracing your System Product Interpreter or EXEC 2 programs or macros.

New CMS Function

DISKID - Obtains information on the physical organization of RESERVEd minidisk.

New CMS Macro Instructions

ABNEXIT - Sets or clears ABEND exit routines.

IMMCMD - Declares, clears, or queries Immediate commands.
WAITECB - Waits on an Event control block (ECB) or a list of ECBs.

New CP Command

PER - Monitor certain events in the user's virtual machine as they occur during program execution.

The System Product Interpreter

"Part 3. Learning to Use EXECs" describes the System Product Interpreter and provides examples of writing programs in the Restructured Extended Executor (REXX) language used with the System Product Interpreter.

The XEDIT PF Keys

Document changes to the XEDIT PF Keys in HELP files and in other files.

512-byte Blocksize

Document support for the 512-byte blocksize for CMS formatted minidisks.

Miscellaneous

This major revision incorporates minor technical and editorial changes.

Summary of Changes
for SC19-6210-1
for VM/SP Release 2

New:

Document changes due to the restructuring of the CMS nucleus.

Support of IOCP and the enhanced ASCII is included.

The following commands and functions are new for Release 2: CMDCALL, DEFAULTS, EXECIO, FILELIST, GLOBALV, IDENTIFY, NAMEFIND, NAMES, NOTE, NUCXDROP, NUCXLOAD, NUCXMAP, PEEK, RDR, RDRLIST, READCARD, RECEIVE, SENDFILE, TELL, and NUCEXT. These commands are documented in the *VM/SP CMS Command Reference*.

Changed:

This major revision incorporates minor technical and editorial changes.

Glossary of Terms and Abbreviations

This section explains or defines the terms, acronyms, and abbreviations that appear in this manual. For a complete list of terms used in VM/SP refer to the *VM/SP Library Guide, Glossary, and Master Index*, GC19-6207. You may also want to refer to the *IBM Vocabulary for Data Processing, Telecommunications, and Office Systems*, GC20-1699.

A

ADCON. Is an A-type address constant used in the calculation of storage addresses.

B

border. A boundary around a window. A user can enter one-letter Border commands from the corners of the border. For example, the letter "P" entered from a border corner pops the window. The border corners are indicated by a " + " (plus) sign.

C

CMS (Conversational Monitor System). A component of Virtual Machine/System Product (VM/SP) that is a conversational operating system designed to run under Control Program (CP).

CMS/DOS. Refers to the functions of CMS that become available when you issue the command: set dos on. CMS/DOS is a part of the normal CMS system and is not a separate system. Users who do not use CMS/DOS are sometimes referred to as OS users, since they use the OS simulation functions of CMS.

CMS files. Refers exclusively to files that are in the fixed block format used by CMS file system commands. VSAM and OS data sets and VSE files are not compatible with the CMS file format and cannot be manipulated using CMS file system commands.

console stack. Refers collectively to the program stack and the terminal input buffer.

CP (Control Program). A component of Virtual Machine/System Product (VM/SP) that controls the resources of the real machine.

D

discontiguous saved segment. An area of storage beyond the address of your virtual machine address space (not contiguous with your virtual storage) where segments are loaded as needed.

disk. Refers to a disk that is in your CMS virtual machine configuration. Also referred to as a virtual disk.

E

ECB. Event Control Block

extended PLIST (untokenized parameter list). Consists of four addresses that indicate the extended form of the command as it was entered at the terminal.

F

FB-512. Refers to the IBM 3370 and 3310 Direct Access Storage Devices.

full-screen CMS. In VM/SP, when a user issues the command SET FULLSCREEN ON, CMS is in a window and can take advantage of 3270-type architecture and windowing support, and various classes of output are routed to a set of default windows. Also, users can type commands almost anywhere on the physical screen and scroll through commands and responses previously displayed. For more information, see windowing.

I

IPL. Initial program load.

L

look-aside entry. A nucleus resident routine becomes a look-aside entry after it has been executed.

M

module. A file whose external references have been resolved.

N

nucleus. That part of CP or CMS that is resident in main storage.

P

| **physical screen.** See screen.

PLIST. Parameter list

program stack. Temporary storage for lines (or files) being exchanged by programs that execute in CMS.

R

REXX language. Restructured Extended Executor language used in System Product Interpreter programs.

S

| **screen.** An illuminated display surface; for example, the display surface of a Cathode Ray Tube (CRT).

SID code. Support Identification code

SCRIPT/VS. A component of the IBM Document Composition Facility program product, which is available from IBM for a license fee. For additional information on SCRIPT/VS usage, see *Document Composition Facility: User's Guide*, SH20-9161.

T

terminal input buffer. Holds lines entered at your terminal until CMS processes them.

tokenized PLIST (parameter list). A string of doubleword aligned parameters occupying successive double words.

V

virtual disk. See disk.

virtual machine. A functional equivalent of a real machine.

| **virtual screen.** In VM/SP, a functional simulation of a physical screen. A virtual screen is a "presentation space" where data is maintained. The user can view pieces of a virtual screen through windows on the physical screen.

VM/SP (Virtual Machine/System Product). A licensed program that controls virtual machines.

| **vscreen.** See virtual screen.

W

| **window.** An area on the physical screen that can be used to display virtual screen data.

| **windowing.** In VM/SP, a set of functions which allow the user to view and manipulate data in user-defined areas of the physical screen called "windows." Windowing support allows the user to define, position, and overlay windows; scroll backward and forward through data; and write data into virtual screens.

Numerics

2305. Refers to the IBM 2305 Fixed Head Storage, Models 1 and 2.

2741. Refers to the IBM 2741 terminal. Information on the 2741 also applies to the IBM 3767 terminal, unless otherwise noted.

3270. Refers to a series of display devices, namely the IBM 3275, 3276 Controller Display Station, and 3277, 3278, and 3279 Display Stations, and the 3290 Information Panel. A specific device type is used only when a distinction is required between device types. Information about display terminal usage also refers to the IBM 3138, 3148, and 3158 Display Consoles when used in display mode, unless otherwise noted.

3284. Refers to the IBM 3284 Printer. Information on the 3284 also pertains to the IBM 3286, 3287, 3288, and 3289 printers, unless otherwise noted.

3310. Refers to the IBM 3310 Direct Access Storage Device.

3330. Refers to the IBM 3330 Disk Storage Models 1, 2, and 11, the IBM 3333 Disk Storage and Control Models 1 and 11, and the IBM 3350 Direct Access Storage in 3330 compatibility mode.

3340. Refers to the IBM 3340 Direct Access Storage Facility and the IBM 3344 Direct Access Storage.

3350. Refers to the IBM 3350 Direct Access Storage device when used in native mode.

3370. Refers to the IBM 3370 Direct Access Storage Device, Models A1, A2, B1, and B2.

3380. Refers to the IBM 3380 Direct Access Storage Device.

3480. Refers to the IBM 3480 Magnetic Tape Subsystem.

370x. Refers to the 3704/3705 Communications Controllers.

3800. Refers to the IBM 3800 Printing Subsystems, Models 1 and 3. A specific device type is used only when a distinction is required between device types.

4248. Refers to the IBM 4248 Printer.

Bibliography

Prerequisite Publications

Virtual Machine/System Product:

Introduction, GC19-6200

Terminal Reference, GC19-6206

A new user of CMS might refer to the *VM/SP CMS Primer*, SC24-5236, for introductory tutorial information on using CMS in display mode. If you are using a line-oriented terminal, then refer to the *VM/SP CMS Primer for Line-Oriented Terminals*, SC24-5242.

Corequisite Publications

Virtual Machine/System Product:

Application Development Guide, SC24-5247

CMS Command Reference, SC19-6209

CP Command Reference, SC19-6211

EXEC 2 Reference, SC24-5219

System Messages and Codes, SC19-6204

System Messages Cross-Reference, SC24-5264

System Product Editor Command and Macro Reference, SC24-5221

System Product Editor User's Guide, SC24-5220

System Product Interpreter Reference, SC24-5239

System Product Interpreter User's Guide, SC24-5238

Quick References

There are publications available as quick reference material when you use VM/SP and CMS. They are:

Virtual Machine/System Product:

Commands (General User) Reference Summary, SX20-4401

Commands (Other than General User) Reference Summary, SX20-4402

EXEC 2 Language, SX24-5124

Quick Reference, SX20-4400

SP Editor Command Language Reference Summary, SX24-5122

System Product Interpreter Language Reference Summary, SX20-5126.

Related VM/SP Publications

Additional descriptions of various CMS functions and commands that are normally used by system support personnel are described in the following publications:

Virtual Machine/System Product:

CMS for System Programming, SC24-5286

CP for System Programming, SC24-5285

Installation Guide, SC24-5237

Operator's Guide, SC19-6202

Planning Guide and Reference, SC19-6201

Virtual Machine:

Diagnosis Guide, LY24-5241

Running Guest Operating Systems, SC19-6212

System Facilities for Programming, SC24-5288

Details on the use of OS/VS EREP operands for the CMS command CPEREP are contained in the *OS/VS, DOS/VSE, VM/370 Environmental Recording, Editing, and Printing Program*, GC28-0772.

Related Publications for OS Users

For information on OS/VS tape label processing, discussed with "Label Processing in OS Simulation" in this publication, refer to:

OS/VS1 Data Management Services Guide, GC26-3874

OS/VS2 MVS Data Management Services Guide, GC26-3875

OS/VS Tape Labels, GC26-3795.

Information on the linkage editor is contained in *OS/VS Linkage Editor and Loader*, GC26-3813.

Related Publications for VSAM and Access Method Services Users

CMS support of Access Method Services is based on VSE and VSE/VSAM. The control statements that you can use are described in *Using VSE/VSAM Command and Macros,* SC24-5144.

Error messages produced by the Access Method Services program, and return codes and reason codes, are listed in *VSE/VSAM Messages and Codes,* SC24-5146.

For a detailed description of VSE/VSAM macros and macro parameters, refer to the *VSE/AF Macro User's Guide,* SC24-5210.

For information on OS/VS VSAM macros, refer to *OS/VS Virtual Storage Access Method (VSAM) Programmer's Guide,* GC26-3838.

Information on formatting virtual minidisks using the Device Support Facility Program is found in the *Device Support Facilities User's Guide and Reference,* GC35-0033.

Related Publications for CMS/DOS Users

The CMS ESERV command invokes the VSE ESERV program, and uses, as input, the control statements that you would use in VSE. These control statements are described in *Guide to the DOS/VSE Assembler,* GC33-4024.

Linkage editor control statements, used when invoking the linkage editor under CMS/DOS, are described in *Guide to the DOS/VSE Assembler*, GC33-4024.

For information on DOS/VSE and CMS/DOS tape label processing, refer to the following publications:

VSE/AF Tape Labels, SC24-5212

VSE/AF Macro User's Guide, SC24-5210.

For information about using DL/I in the CMS/DOS environment, see *DL/I DOS/VS Data Base Administration*, SH24-5011.

The VM/SP Library (Part 1 of 3)

Evaluation

- General Information GC20-1838
- Introduction GC19-6200

Index

- Library Guide, Glossary, and Master Index GC19-6207

Planning

- Planning Guide and Reference SC19-6201
- Running Guest Operating Systems GC19-6212
- Release 5 Guide SC24-5290
- Distributed Data Processing Guide SC24-5241

Installation

- Installation Guide SC24-5237

Applications

- Application Development Guide SC24-5247
- Programmer's Guide to the SRPI for VM/SP SC24-5291

Operation

- Operator's Guide SC19-6202

Reference Summaries

To order all of the Reference Summaries, use order number SBOF-3242

- Commands (General User) SX20-4401
- Commands (Other than General User) SX20-4402
- SP Editor Command Reference Summary SX24-5122
- EXEC 2 Reference Summary SX24-5124
- Sys.Prod Interpreter Reference Summary SX24-5126
- CMS Primer Summary of Commands SX24-5151
- CMS Primer Line-Oriented Summary of Commands SX24-5159
- Problem Reporting Summary (Poster) SX24-5171
- Summary of End Use Tasks and Commands (Poster) SX24-5173

The VM/SP Library (Part 2 of 3)

End Use

- Terminal Reference GC19-6206
- CMS Primer SC24-5236
- CMS Primer for Line-Oriented Terminals SC24-5242
- CMS User's Guide SC19-6210
- CMS Command Reference SC19-6209
- CMS Macros and Functions Reference SC24-5284
- System Product Editor User's Guide SC24-5220
- System Product Editor Command and Macro Reference SC24-5221
- System Product Interpreter User's Guide SC24-5238
- System Product Interpreter Reference SC24-5239
- EXEC 2 Reference SC24-5219
- CP Command Reference SC19-6211
- Quick Reference SX20-4400

Diagnosis

- System Messages and Codes SC19-6204
- System Messages Cross-Reference SC24-5264
- Service Routines Program Logic LY20-0890
- Problem Reporting Guide SC24-5282
- VM Diagnosis Guide LY24-5241
- GCS Diagnosis Reference LY24-5239
- Problem Determination Vol. 1 (CP) LY20-0892
- Data Areas and Control Blocks Vol. 1 (CP) LY24-5220
- Problem Determination Vol. 2 (CMS) LY20-0893
- Data Areas and Control Blocks Vol. 2 (CMS) LY24-5221
- OLTSEP and Error Recording Guide SC19-6205
- VM Problem Determination Reference Information LX23-0347
- VM CP Internal Trace Table (Poster) LX24-5202

The VM/SP Library (Part 3 of 3)

Administration

VM System Facilities for Programming
SC24-5288

CP for System Programming
SC24-5285

CMS for System Programming
SC24-5286

TSAF Reference
SC24-5287

GCS Command and Macro Reference
SC24-5250

Auxiliary Communication Support

VTAM Installation and Resource Definition
SC23-0111

VTAM Customization
SC23-0112

VTAM Operation
SC23-0113

VTAM Messages and Codes
SC23-0114

VTAM Reference Summary
SC23-0135

VTAM Programming
SC23-0115

VTAM Diagnosis Guide
SC23-0116

VTAM Diagnosis Reference
LY30-5582

VTAM Data Areas (VM)
LY30-5583

RSCS Networking Version 2 General Information
GH24-5055

RSCS Networking Version 2 Planning and Installation
SH24-5057

RSCS Networking Version 2 Operation and Use
SH24-5058

RSCS Networking Version 2 Diagnosis Reference
LY24-5228

RSCS Networking Version 2 Ref. Summary
SX24-5135

VM/Pass-Through Facility General Information
GC24-5206

VM/Pass-Through Facility Guide and Reference
SC24-5208

VM/Pass-Through Facility Logic
LY24-5208

Index

Special Characters

¢ logical line delete symbol 295
.BX (BOX) format word 282
.CM (COMMENT) format word 284
.CS (CONDITIONAL SECTION) format word 285
.FO (FORMAT MODE) format word 285
.IL (INDENT LINE) format word 286
.IN (INDENT) format word 287
.MT (MENU TYPE) format word 288
.OF (OFFSET) format word 289
.SP (SPACE LINES) format word 290
.TR (TRANSLATE CHARACTER) format word 290
&PUNCH control statement
  punching jobs to CMS batch facility 214
&TRACE statement in EXEC 2 224
$HLPABBR filetype usage in CMS 52
$HLPCMS filetype usage in CMS 52
$HLPCMSQ filetype usage in CMS 52
$HLPCMSS filetype usage in CMS 52
$HLPCP filetype usage in CMS 52
$HLPCPQU filetype usage in CMS 52
$HLPCPSE filetype usage in CMS 52
$HLPDEBU filetype usage in CMS 52
$HLPEDIT filetype usage in CMS 52
$HLPEXC2 filetype usage in CMS 52
$HLPEXEC filetype usage in CMS 52
$HLPGROU filetype usage in CMS 52
$HLPHELP filetype usage in CMS 52
$HLPIPCS filetype usage in CMS 52
$HLPMENU filetype usage in CMS 52
$HLPMSG filetype usage in CMS 52
$HLPPREF filetype usage in CMS 52
$HLPQUER filetype usage in CMS 52
$HLPREXX filetype usage in CMS 52
$HLPSET filetype usage in CMS 52
$HLPSQLD filetype usage in CMS 52
$HLPTASK filetype usage in CMS 52
$HLPTSAF filetype usage in CMS 52
$HLPXEDI filetype usage in CMS 52
* (asterisk)
  as fileids on command lines 49
  in FILELIST command 71
  in filemode field 61
  in LISTFILE command 72
*/ (end of a REXX comment statement) 225
/* (slash asterisk)
  CMS batch facility control card used to signal end of job 208
  end-of-file indicator in batch job 218
  in REXX language interpreted as comment 225
/JOB control card description 207
/SET control card description 208
# (logical line end character)
  description 295
  using when setting PF (program function) keys 12
#CP command 313
#CP function
  used when setting PFnn RETRIEVE 11
  using on display terminals 11
@ (logical character delete character) 295
  using when setting PF (program function) keys 13
= (equal sign)
  entered in fileids on command lines 49
" (logical escape character) 295

A

A-disk 57
ACCESS command
  accessing CMS disks 26
access method services
  See CMSPROG
accessing
  disks
    as read-only extensions 59
    in CMS 26
    in CMS batch virtual machine 210
  file directories for CMS disks 66
accessing DOS and OS disks
  See CMSPROG
ACNT command 313
ADD operand of MACLIB command
  See CMSPROG
adding
  BRIEF section to HELP files 276
  DESCRIPT section to HELP files 276
  DETAIL section to HELP files 276
  ERRORS section to HELP files 277
  FORMAT section to HELP files 276
  HELP components 279
  NOTES section to HELP files 277
  OPTIONS section to HELP files 277
  PARMS section to HELP files 277
  RELATED section to HELP files 277

These symbols are used in the index to refer to other VM and VM/SP books:
CMSPROG—VM/SP CMS for System Programming, SC24-5286
DIAG—VM Diagnosis Guide, LY24-5241

address
  stops
    to enter CP environment 37
  virtual
    for unit record devices 119
ADSTOP command 313
ALARM VSCREEN command 302, 322
ALL option of DETAIL HELP 94
allocating space for VSAM files
  See CMSPROG
altering
  characteristics of spool files 122
altering existing HELP files 292
AMSERV command 53
  See also CMSPROG
APL used on display terminal 42
applications in full-screen CMS 334
ASM3705 filetype usage in CMS/DOS 51
ASSEMBLE command usage in CMS 51
assembler language macros
  See CMSPROG
assembling programs
  See CMSPROG
ASSGN command
  See CMSPROG
assigning filemode letters to disks 57
ATTACH command 314
attention interruptions
  causing 38
  virtual machine 40
AUTOLOG command 314
automatic
  IPL 9
  save function for editors 111
AUTOREAD operand of CMS SET command for display terminals 15
AUXPROC option of FILEDEF command
  See CMSPROG
AUXxxxx filetype
  usage in CMS 51

B

BACKSPAC command 314
batch facility
  See batch facility
batch jobs
  for CMS batch facility 205
  for Non-CMS users 217
  purging 213
  reordering 213
batch processing in CMS 205
BDAM access method
  See CMSPROG
BEGIN command 314
BEGIN command to return to virtual machine environment 33
beginning
  virtual machine execution 33
  your terminal session 6
BLOCK option of FILEDEF command
  See CMSPROG
BLP
  See bypass label processing, tapes
Border commands
  scrolling forward and backward 194
  scrolling right and left 197
BPAM access method
  See CMSPROG
BRIEF HELP 93
BSAM access method
  See CMSPROG
BUFSP option of DLBL command
  See CMSPROG
bypass label processing, tapes 134

C

caller id in tape label processing 136
canceling
  changes during editing session 111
card punch
  used to send jobs to CMS batch facility 205
card reader
  restriction on use in job for CMS batch facility 212
  spooling punch or printer files to 122
cards
  /* as end-of-file indicator 207
  as input to CMS batch facility 206
CAT option of DLBL command
  See CMSPROG
CATCHECK command
  See CMSPROG
causing breaks in text 290
CHANGE command 314
  used to change hold status on spool files 123
changing
  characteristics of spool files 119
  characteristics of unit record devices 119
  filemode numbers 64
  output representation of a character 290
  the HELP facility 267
channel address word
  See DIAG
channel status word
  See DIAG
character
  deleting from line 295
  special
    valid in CMS file identifiers 47
CLASS operand of SPOOL command 120
classes
  CP command privilege 311
  of CP SPOOL files 120
cleanup functions for VSAM 224

CLEAR operand of DLBL command
  See CMSPROG
CLEAR VSCREEN command 302, 322
CLEAR WINDOW command 302, 322
clock indicator in full-screen CMS 177
CLOSE command 314
CLOSE macro for OS simulation
  See CMSPROG
CMS (Conversational Monitor System)
  considerations for migrating to full-screen CMS 330
  description 3
  file system 47
  files
    See files, CMS
  loading into your virtual machine 9
  migrating to full-screen CMS 330
  understanding it 3
CMS (Conversational Monitor System) commands
  execution characteristics 68
  general information 4
  processing tape labels 143
  search order 67
  summary 299
  transient area 68
  used with EXECS written in REXX language 233
  user area 68
CMS batch facility
  /* 207
  /JOB 207
  /SET 208
  control cards 206
  controlling spool files 211
  description 205
  housekeeping done after executing job 210
  how jobs are processed 209
  ID card 206
  jobs for non-CMS users 217
  using EXEC procedure to submit jobs 214
CMS Editor
  environment 35
CMS environment 34
CMS EXEC procedures
  submitting jobs to CMS batch facility 206
CMS EXEC procedures to execute OS programs
  See CMSPROG
CMS files
  See files
CMS in full-screen
  See full-screen CMS
CMS macro libraries
  See CMSPROG
CMS subset
  See also CMSPROG
  environment 35
  using 117
CMS/DOS
  end-of-tape processing 146
  overview 38
  tape label processing 140
CMS/DOS commands
  See CMSPROG
CMSLIB MACLIB
  See CMSPROG
CMSPROG
  See VM/SP CMS for System Programming
CMSSERV command 302
CMSUT1 file and CMS commands that create 56
CNTRL filetype
  usage in CMS 51
command abbreviations used in HELP 279
command HELP 92, 98
command line in full-screen CMS 177
commands
  environments 31
  how to enter 3
  language 3
    CMS 4
    CP 4
  search order 67
  translation synonyms 76
  translations 76
COMMANDS command 314
COMMENT statement
  See CMSPROG
comments
  in HELP text files 284
  in REXX language 225
communicating
  with other computer users 155
  with VM/SP 3
COMP operand of MACLIB command
  See CMSPROG
COMPACT option of DDR command 127
compacting tape output 127
COMPARE command to compare contents of two CMS files 73
compilers supported in CMS 4
component MENU in HELP 100
components of VM/SP 3
composing notes 163
CONCAT option of FILEDEF command
  See CMSPROG
CONCEAL option of SET command 35
conditionally displaying text 285
considerations for windowing and full-screen CMS 321
console
  log
    creating disk file from 27
    printing 27
    produced by CMS batch facility 212
  output spooling for display terminal 27
CONT
  operand, of CP SPOOL command 121
continuous spooling 121
control cards for CMS batch facility

These symbols are used in the index to refer to other VM and VM/SP books:
CMSPROG—VM/SP CMS for System Programming, SC24-5286
DIAG—VM Diagnosis Guide, LY24-5241

See CMS batch facility
Control Program (CP)
basic description 3
command information 4
considerations for migrating to full-screen CMS 332
environment, entering 33
privilege classes 311
spooling facilities 119
controlling
terminal output 83
conventions, notational for HELP 276
Conversational Monitor System (CMS)
considerations for migrating to full-screen CMS 330
description 3
file system 47
loading into your virtual machine 9
migrating to full-screen CMS 330
understanding it 3
CONVERT COMMANDS command 302
converting
file to a HELP file 281
fixed-length file to variable-length format 74
variable-length file to fixed-length format 74
COPY
filetype
usage in CMS 51
function on display terminals 29
operand of CP SPOOL command 121
COPYFILE command
changing filemode numbers 64
copying files from one virtual disk to another 73
creating small files from large one 116
used to change record formats of files 74
copying
contents of a display screen 29
file, with COPYFILE command 73
files, from one device to another 125
from tape to disk 125
spool files 121
corrections
of lines as you enter them 11
using logical line editing symbols 295
COUPLE command 314
CP (Control Program)
basic description 3
command information 4
considerations for migrating to full-screen CMS 332
environment, entering 33
privilege classes 311
spooling facilities 119
CP (Control Program) commands
summary 313
used in job for CMS batch facility 212
used in System Product Interpreter EXECs 228
CP command 314
CP READ status on display terminal 14
CPTRAP command 314
creating
command HELP files 275
file with System Product Editor 109
HELP text files 291
HELPMENU files 271
HELPTASK files 274
menus, HELP file 271
notes 163
one spool file from many files being printed or punched 121
PROFILE EXEC 229
System Product Interpreter EXECs 225
CURSOR VSCREEN command 302, 322

D

Data Control Block (DCB)
See CMSPROG
DCP command 314
DCSSGEN command 308
DCSSMAP filetype usage in CMS 51
DDR command
COMPACT option for tape output 127
DDR program used to dump to tape 127
DEBUG
command 37
subcommands 37
debug environment 37
debugging commands and subcommands
See DIAG
default
logical line editing symbols 295
notebook 165
setting with DEFAULTS command 165
DEFINE command 314
defining temporary disks 24
to increase virtual storage size 115
DEFINE VSCREEN command 248, 302, 322
DEFINE WINDOW command 248, 302, 322
defining
logical line editing symbols 297
tapes
nonstandard 140
standard 140
unlabeled 140
temporary disks 24
Definition Language for Command Syntax (DLCS) file 75
DEL operand of MACLIB command
See CMSPROG
DELETE VSCREEN command 302, 322
DELETE WINDOW command 303, 322
deleting HELP files 292
density of tapes 151
DESCRIPT option of DETAIL HELP 94
DETACH command 315
DETACH command after RELEASE command 27
detaching disks 27

without releasing them 66
DETAIL HELP 94
DIAG
See VM Diagnosis Guide
DIAL command 315
DIRECT filetype usage in CMS 51
DISABLE command 315
DISCONN command 315
DISCONN, command 10
disconnecting your terminal from your virtual machine 10
Discontiguous Shared Segment (DCSS) 230, 237
discontinuing access to EXEC in Discontiguous Shared Segment (DCSS) 238
DISK command
DUMP operand, using 125
LOAD operand restricted in job for CMS batch facility 212
LOAD operand, using 125
loading files 164
disk determination
default for reading files
commands that search all accessed disks 60
commands that search only A-disk 60
commands that search only A-disk and its extensions 60
default for writing files
commands for which you must specify filemode 61
commands that write output to A-disk 62
commands that write output to read/write disk 62
DISKMAP command 308
disks
defined in VM/SP directory entry 23
defining temporary disks for terminal session 24
definition 23
extensions 58
full, in editing session, recovery from 116
how much space is on a disk 65
identifying to CMS 26
linking 26
listing information about the files on 70
master file directory 66
providing for CMS batch virtual machine 210
query status of 65
read-only extensions 58
search order 26
sharing 25
DISP MOD option of FILEDEF command
See CMSPROG
display
full screen in System Product Editor 112
multiple views 109
DISPLAY command 315
See also DIAG
display screen status notices 14
display terminals
characteristics 14
entering commands 10
example of display screen 30
Extended highlight feature 18
line-mode 117
retrieving previously entered data 11
setting PF keys 12
signalling interruptions 38
displaying
list of CMS files 70
screens of data 16
disposition of spool files 119
DL/I programs in CMS/DOS
See CMSPROG
DLBL command
assigning filemode numbers 64
DLCS (Definition Language for Command Syntax) file 75
DMCP command 315
DMSSP MACLIB
See CMSPROG
DMSVSR command 224
DOS (Disk Operating System)
See CMSPROG
DOSLIB command
See CMSPROG
DOSLKED command
See CMSPROG
DOSMACRO MACLIB
See CMSPROG
DOSPART operand of CMS SET command
See CMSPROG
DRAIN command 315
drawing boxes 282
DROP WINDOW command 187, 303, 322
DSERV command
See CMSPROG
DSN operand of DLBL command
See CMSPROG
DSORG option of FILEDEF command
See CMSPROG
DUMP command 315
See also DIAG
duplicating
filenames or filetypes 48
DVOL1 function in tape command processing 144

E

ECHO command 315
EDIT command
edit mode, returning from input mode 112
editing
CMS files 109
session 109
with logical line editing symbols 295

These symbols are used in the index to refer to other VM and VM/SP books:
CMSPROG—VM/SP CMS for System Programming, SC24-5286
DIAG—VM Diagnosis Guide, LY24-5241

ENABLE command 315
end of file
  indicating for input stream to batch virtual machine 208
end-of-tape, processing 146
end-of-volume, processing 146
ending
  editing session 22
  terminal session 10
entering
  APL characters on display terminal 42
  CMS commands, in CMS subset environment 36
  CMS environment 9
  CMS/DOS environment 38
  commands
    from the screen 190
    more than one on command line 295
    on display terminal 10
  CP commands
    from CMS command environment 33
    from edit environment 36
  CP environment
    from CMS environment 33
  debug environment
    via CP EXTERNAL command 37
    via DEBUG command 37
  DEBUG subcommands 37
  HELP facility 5
  Immediate commands 38
  logical line editing symbols as data 297
  null lines 3
environments of VM/SP 31
ERASE command 23
erasing
  CMS files 23
  files to clear disk space during editing session 117
error messages
  controlling whether you receive them 4
error processing
  messages 149
  NSL routines 149
  OS simulation 149
  standard label processing 149
ERRORS option of DETAIL HELP 94
ESERV command
  See CMSPROG
EXEC
  filetype used in CMS 51
  loading into Discontiguous Shared Segment (DCSS) 230
  loading into storage 237
  procedures
    using to submit jobs to CMS batch facility 215
EXEC 2
  &TRACE statement 225
  comparison to EXEC 221
  comparison to System Product Interpreter 221
  files
    attributes 225
    format 225
  invoking 226
  used with System Product Editor 110
EXECDROP command
  description 233
  discontinuing access to EXEC in Discontiguous Shared Segment (DCSS) 238
  example 237
EXECIO command
  description 233
  example 234
EXECLOAD command
  description 233
  example 236
  improving performance of HELP 91
EXECMAP command
  description 233
  example 237
  listing EXECs in storage 238
EXECSTAT command
  description 233
  example 237
executing
  command using program function (PF) keys 11
  EXEC procedures 226
  Immediate commands in EXECs 233
  PROFILE EXEC 229
execution characteristics of CMS commands 68
execution state indicator in full-screen CMS 177
exiting from
  editing session 22
  execution of a program 40
  terminal session 10
EXTEND option of DETAIL HELP 96
extensions, read-only, using 58
EXTENT option of DLBL command
  See CMSPROG
EXTERNAL command 315
  See also DIAG

F

FETCH command
  See CMSPROG
file
  converting to a HELP file 281
  directories, CMS 66
  identifier
    changing with SAVE subcommand 111
    CMS, rules for assigning 47
    coded as asterisk (*) 49
    coded as equal sign (=) 49
  system 47
file manipulation using System Product Editor 109
file status table (FST)
FILE subcommand to write files to disk 109
FILEDEF command

assigning filemode numbers 64
OS simulation 131
standard tape labels 131
tape label processing 137
FILELIST command used to list disk files in full screen environment 70
filemode
extensions 58
in file identifier 47
letters
assigning 48
when to specify for reading files 60
when to specify for writing files 61
numbers
descriptions 57
when to specify 62
filemode 0 62
filemode 1 63
filemode 2 63
filemode 3 63
filemode 4 63
filemode 5 64
filemode 6 64
filename 47
files
CMS
erasing 72
format 47
identifiers 47
renaming 73
discarding after being read 72
logical grouping 55
private 62
requesting information about 70
shared by users 62
splitting into smaller files 116
too large to edit, what to do 116
filetype
created by assembler and language processors 50
HELP facility 268
$HLPABBR 52
$HLPCMS 52
$HLPCMSQ 52
$HLPCMSS 52
$HLPCP 52
$HLPCPQU 52
$HLPCPSE 52
$HLPDEBU 52
$HLPEDIT 52
$HLPEXC2 52
$HLPEXEC 52
$HLPGROU 52
$HLPHELP 52
$HLPIPCS 52
$HLPMENU 52
$HLPMSG 52
$HLPPREF 52
$HLPQUER 52
$HLPREXX 52
$HLPSET 52
$HLPSQLD 52
$HLPTASK 52
$HLPTSAF 52
$HLPXEDI 52
HELPABBR 52
HELPCMS 52
HELPCMSQ 52
HELPCMSS 52
HELPCP 52
HELPCPQU 52
HELPCPSE 52
HELPDEBU 52
HELPEDIT 52
HELPEXC2 52
HELPEXEC 52
HELPGROU 52
HELPHELP 52
HELPIPCS 52
HELPMENU 52
HELPMSG 52
HELPPREF 52
HELPQUER 52
HELPREXX 52
HELPSET 52
HELPSQLD 52
HELPTASK 52
HELPTSAF 52
HELPXEDI 52
in file identifier 47
reserved for language processors 50
temporary work files 56
used by CMS commands 51
FLUSH command 315
FOR operand of CP SPOOL command, usage 121
FORCE command 315
FORMAT command to format a CMS disk 24
FORMAT option of DETAIL HELP 94
format words
.BX (BOX) 282
.CM (COMMENT) 284
.CS (CONDITIONAL SECTION) 285
.FO (FORMAT MODE) 285
.IL (INDENT LINE) 286
.IN (INDENT) 287
.MT (MENU TYPE) 288
.OF (OFFSET) 289
.SP (SPACE LINES) 290
.TR (TRANSLATE CHARACTER) 290
summary 282
format-mode, processing 292
formatting
CMS disks, example 24
FREE command 315
FSCB (file system control block)
See CMSPROG
FSCLOSE macro
See CMSPROG

These symbols are used in the index to refer to other VM and VM/SP books:
CMSPROG—VM/SP CMS for System Programming, SC24-5286
DIAG—VM Diagnosis Guide, LY24-5241

FSERASE macro
  See CMSPROG
FSREAD macro
  See CMSPROG
FSWRITE macro
  See CMSPROG
full disk during editing session 116
full screen display, with System Product Editor 109
full-screen CMS 175
  Border commands 192, 323
  commands 321
  considerations for migrating 330
  entering commands from the screen 190
  location information 178
  maximizing windows 253
  messages 186
  migrating from line-mode CMS 330
  PA and PF keys
    Backward 182, 327
    Clear Top 182, 327
    Cursor 182, 327
    Filelist 182, 327
    Forward 182, 327
    Help 182, 327
    Left 182, 327
    Pop Msg 182, 327
    Quit 182, 327
    Rdrlist 182, 327
    Retrieve 182, 327
    Right 182, 327
  PA1 key 185
  physical screen characteristics
    command line 177
    location information 176
    PF key definition area 177
    scrollable data area 176
    status area 177
    title line 176
  positioning windows 251
  restoring windows 254
  setting the reserved line 257
  setting window borders 255
  sizing windows 252
  status information 177
  status notices
    enter a command or press a PA or PF key 17
    enter your response in vscreen vname 17
    executing a command 17
    scroll forward for more information in vscreen vname 17
  system-defined windows and virtual screens 179
  using 175
function types, CP command 311

G

GCS filetype used in CMS 51
GEN operand of MACLIB command
  See CMSPROG
GENMOD command
  See CMSPROG
GENMSG command 53, 304
GENTSAF command 308
GET subcommand of XEDIT
  creating small files from large one 116
GET VSCREEN command 304, 322
GLOBAL command
  See CMSPROG
GLOBALV command used with REXX language System Product Interpreter EXECs 239
GLOBALV filetype usage in CMS 51
GROUP filetype use in CMS 51

H

HALT command 316
halting
  program execution 40
  screen status 16
  System Product Interpreter EXECs 234
  terminal displays 41
HDR1 tape label 145
HELP
  command
    BRIEF 93
    DETAIL 94
    RELATED 95
  format words
    summary 282
  menus
    component 100
    task 99
  message 98
  notational conventions 276
HELP Facility
  components 91, 279
  filetypes 268
  keys, PF and PA2 103
  naming conventions 267
  tailoring 267
  using 91
  using System Product Editor 104
HELP file
  adding 291
  altering existing 292
  BRIEF section, adding 276
  changing menus 292
  converting to a formatted HELP file 281
  creating a new file 291
  deleting 292

DESCRIPT section, adding 276
DETAIL section, adding 276
ERRORS section, adding 277
FORMAT section, adding 276
how to name 267
NOTES section, adding 277
OPTIONS section, adding 277
PARMS section, adding 277
printing 103, 105
RELATED section, adding 277
HELPABBR filetype usage in CMS 51
HELPCMS filetype usage in CMS 51
HELPCMSQ filetype usage in CMS 51
HELPCMSS filetype usage in CMS 51
HELPCONV command
description 281
how it treats comments 285
summary of format words 282
using to create additional HELP files 280
HELPCP filetype usage in CMS 51
HELPCPQU filetype usage in CMS 51
HELPCPSE filetype usage in CMS 51
HELPDEBU filetype usage in CMS 51
HELPEDIT filetype usage in CMS 51
HELPEXC2 filetype usage in CMS 51
HELPEXEC filetype usage in CMS 51
HELPGROU filetype usage in CMS 51
HELPHELP filetype usage in CMS 51
HELPIPCS filetype usage in CMS 51
HELPMENU files, creating 271
HELPMENU filetype usage in CMS 51
HELPMSG filetype usage in CMS 51
HELPPREF filetype usage in CMS 51
HELPQUER filetype usage in CMS 51
HELPREXX filetype usage in CMS 51
HELPSET filetype usage in CMS 51
HELPSQLD filetype usage in CMS 51
HELPTASK files, creating 274
HELPTASK filetype usage in CMS 51
HELPTSAF filetype usage in CMS 51
HELPXEDI filetype usage in CMS 51
HI (Halt Interpretation) Immediate command 234
HIDE WINDOW command 304, 322
highlighting words in a file using HELP 278
HOLD command 316
HOLD operand of SPOOL command 120
holding
display on terminal 16
spool file to keep them from being processed 120
HOLDING screen status 16
HT (Halt Type) Immediate command 41
HX (Halt Execution) Immediate command
effect in CMS subset 36

I

ID card to submit job to CMS batch facility 206
IEBPTPCH utility program to create CMS files from tapes created by 150
IEBUPDTE utility program to create CMS files from tapes created by 151
IEHMOVE utility program to create CMS files from tapes created by 151
IJSYSCT, defining in CMS/DOS
See CMSPROG
IJSYSUC, defining in CMS/DOS
See CMSPROG
Immediate commands
entering on display terminal 38
using with System Product Interpreter programs 234
IMPCP operand of CMS SET command, setting 34
IMPEX operand of CMS SET command, usage 226
implied
CP function, SET IMPCP, usage 34
EXEC function, SET IMPEX, usage 226
IMPORT access method services function
See CMSPROG
INCLUDE command
See CMSPROG
increasing virtual machine storage 115
indenting text 286
INDICATE command 316
initial program load (IPL)
entering CMS environment 9
linking Installation Discontiguous Shared Segment (DCSS) 238
input mode
on display terminal in line-mode 117
when in full screen XEDIT environment 112
INPUT subcommand
used to enter data in XEDIT 112
INSERT statement
See CMSPROG
inserting lines in a file being edited 112
Installation Discontiguous Shared Segment (DCSS)
discontinuing 238
use of 230, 237
using IPL command to link 238
interrupting
program execution 38
interruptions
signaling on display terminal 38
invoking
System Product Editor 109
System Product Interpreter EXECs 226
VSAPL on display terminal 42
IPL (initial program load)
entering CMS environment 9
linking Installation Discontiguous Shared Segment (DCSS) 238

These symbols are used in the index to refer to other VM and VM/SP books:
CMSPROG—VM/SP CMS for System Programming, SC24-5286
DIAG—VM Diagnosis Guide, LY24-5241

IPL command 316
ISAM access method
  See CMSPROG
ITASK command 308

J

jobname for job sent to CMS batch facility
  specifying 207
  used to identify spool files 212
jobs for CMS batch facility, submitting 205

K

keywords, translations of 76

L

label off processing, tapes 135
label processing, general description 131
LABELDEF command
  description 145
  in CMS/DOS tape label processing 141
  in tape processing 143
  standard labels 132
  use of 145
labels
  writing on CMS disks 24
labels for DOS, VSE/VSAM, and OS/VSAM disks
  See CMSPROG
LABOFF (label off) processing 135
LANGGCTL filetype usage in CMS 52
LANGGEN command 308
LANGMAP filetype usage in CMS 53
LANGMCTL filetype usage in CMS 53
LANGMERG command 308
language statements
  in EXEC 2 language 223
  in REXX language, for System Product Interpreter 221
large files split into smaller files 116
leaving
  CMS subset environment 36
  CMS/DOS environment 38
  debug environment 37
  input mode 113
  XEDIT environment 109
length of CMS ready message, changing 19
libraries for CMS, DOS, and DOS/VSE
  See CMSPROG
line-mode
  migrating to full-screen CMS 330
  using the editor 117
LINEDIT macro
  See CMSPROG
lines, deleting at terminal before entering 296
LINK command 316
  format, in job for CMS batch facility 211
  linking to other user's disks 25
link-editing programs in CMS/DOS
  See CMSPROG
linking
  Installation Discontiguous Shared Segment (DCSS) 238
  to other user's disks 25
  to your own disks 26
LISTCAT access method services function
  See CMSPROG
LISTCPDS filetype usage in CMS 53
LISTCRA access method services function
  See CMSPROG
LISTDS command
  See CMSPROG
LISTFILE command used to list your disk files 70
listing
  information
    about CMS files 70
    about disks 21
    about your terminal 297
    about your virtual machine 161
    requested 81
  members of MACLIB
LISTING files
  created by assembler and language processors 53
LISTING filetype
  usage in CMS 53
LISTIO command
  See CMSPROG
LIST38PP filetype usage in CMS 53
LIST3800 filetype usage in CMS 53
LIST3820 filetype usage in CMS 53
LKED command
  See CMSPROG
LKEDIT filetype usage in CMS 53
LOADBUF command 316
loading
  CMS into your virtual machine 9
  EXECs into storage 237
LOADLIB filetype usage in CMS 53
LOADMOD command
  See CMSPROG
LOADVFCB command 316
LOCATE command 316
location information in full-screen CMS 176, 178
LOCK command 316
locking the terminal keyboard 10
logging of messages 191
logging of warnings 191
logging off VM/SP 10
logging on to VM/SP
  at a 3270-type terminal 7
  at other types of terminals 8

logical
  character delete symbol 295
  escape symbol 296
  line delete symbol 296
  line editing symbols
    defining 297
    overriding 297
  line end symbol 295
logo screen, logging on from 7
LOGOFF command 10, 316
LOGON command 316
LOGON command to contact VM/SP 8
LRECL option of COPYFILE command 74

M

MACLIB command
  See CMSPROG
MACLIB filetype usage in CMS 54
MACLIST command
  See CMSPROG
MACRO filetype usage in CMS 54
macro libraries
  See CMSPROG
map filetype usage in CMS 54
MAP operand of MACLIB command
  See CMSPROG
master file directory 66
MAXIMIZE WINDOW command 253, 305, 322
MEMBER option of FILEDEF command
  See CMSPROG
MEMO filetype 57
  for documentation 57
menu
  changing 292
  component 100
  creating 271
  example, of creation 273
  task 99, 274
message class indicator in full-screen CMS 177
MESSAGE command 316
message HELP 98
MESSAGE LOGFILE 191
MESSAGE window
  dropping 187
  popping 187
  using to edit names file 188
  viewing messages 186
messages
  controlling whether you receive them 4
  displayed when logging on 9
  from CMS batch facility 209
  in full-screen CMS 186
  logging 191
  sending to other virtual machine users 160
  when display screen is full 16
MIGRATE command 316
migration to full-screen CMS 330
minidisk, definition of 1
MINIMIZE WINDOW command 305, 322
mode
  edit and input 22
  setting with CP TERMINAL command 39
  switching 31
modifying CMS files, commands to use 88
MODULE filetype usage in CMS 54
MONITOR command 317
MORE... status on display screen 16
MOREHELP command 305
MOVEFILE command
  copying CMS files from tapes created by 127
  copying tape files 150
  description 145
  reading files from virtual card reader 125
  use of 145
moving CMS files, commands to use 89
MSGNOH command 317
MULT option of DLBL command
  See CMSPROG
multi-volume tape processing
  See end-of-volume, processing

N

names file edited with MESSAGE window 188
NAMES filetype usage in CMS 54
naming
  CMS files 47
  conventions for HELP files 267
  user commands 67
NETLOG filetype use in CMS 54
NETWORK command 317
NL processing for tapes
  See no label processing
no label processing 131, 134
NOCLEAR option of XEDIT command used in EXEC procedure 118
nonstandard label processing, tapes 135
nonstandard label routine, writing 135
nonstandard labelled tapes, defining 140
NOPROF option of ACCESS command to suppress execution of PROFILE EXEC 229
NOT ACCEPTED status on display screen 17
notational conventions for HELP 276
NOTEBOOK filetype use in CMS 54
NOTES option of DETAIL HELP 94
NOTREADY command 317
NSL (nonstandard label) processing 135
NUCXLOAD command 53, 54, 300, 305
null line
  entering to determine environment 32
  input data from terminal 3

These symbols are used in the index to refer to other VM and VM/SP books:
CMSPROG—VM/SP CMS for System Programming, SC24-5286
DIAG—VM Diagnosis Guide, LY24-5241

to resume program execution after attention interruption 41

O

offsetting text 289
OPEN macros OS simulation 131
OPTIONS option of DETAIL HELP 94
ORDER command 317
  description 317
  selecting files for processing 123
OS (Operating System)
  cleanup 224
  simulation, end-of-tape processing 146
  utility programs to create CMS files from tapes created by 150
OSMACRO MACLIB
  See CMSPROG
OSMACRO1 MACLIB
  See CMSPROG
OSVSAM MACLIB
  See CMSPROG
output
  from CMS batch facility 212
  from virtual console, spooling 28

P

parameter lists
  See CMSPROG
parent disk, of read-only extension 58
PARMS option of DETAIL HELP 94
PARSECMD command 305
passing
  global variables between EXECs with GLOBALV 239
password
  for your virtual machine 6
  supplying on LINK command line 25
  suppression on command line 25
PA1 key in full-screen CMS 185
PA1 key to enter CP environment 16
PA2 key in HELP 103
PDS option of MOVEFILE command
  See CMSPROG
PER command 317
  See also DIAG
PERM option of FILEDEF command
  See CMSPROG
PF keys
  See program function (PF) key
POP WINDOW command 187, 305, 322
POSITION WINDOW command 251, 305, 322
preparing jobs for CMS batch facility 210
PRINT command to print CMS files 55
printer files
  produced by job running in batch virtual machine 211
  querying status of 123
  spooling 122
printing
  CMS files 55
  HELP files 105
  multiple copies 121
PRINTL macro
  See CMSPROG
privilege classes for CP commands 311
processing, tapes
  BLP 134
  LABOFF 135
  NL 134
  NSL 135
PROFILE EXEC
  creating 229
  description 229
  sample, using REXX language 229
  suppressing execution 229
  use to make EXECS storage-resident 230
program function (PF) key
  ? 103
  All 103
  Backward 103
  Brief 103, 104
  Clocate 103
  Cursor 104
  Forward 103
  Help 103
  Morehelp 104
  PF Key 103
  Print 103
  Quit 103
  Related 104
  Return 103
  toggling between keys 101
  Top 103
program function (PF) keys
  definition area in full-screen CMS 177
  in full-screen CMS
    Backward 182, 327
    Clear Top 182, 327
    Cursor 182, 327
    Filelist 182, 327
    Forward 182, 327
    Help 182, 327
    Left 182, 327
    Pop Msg 182, 327
    Quit 182, 327
    Rdrlist 182, 327
    Right 182, 327
  in WM environment
    Backward 186, 329
    Clear 186, 329
    Copy 186, 329
    Forward 186, 329
    Help 186, 329

Left 186, 329
Maximize 186, 329
Quit 186, 329
Restore 186, 329
Retrieve 186, 329
Right 186, 329
Top 186, 329
setting 12, 184
COPY function 29
in PROFILE EXEC 229
to retrieve previous line entered 11
using 12
using in FILELIST 70
using to send notes 164
using when composing a note 163
using when receiving files when in RDRLIST 169
programs
written in REXX language for System Product Interpreter 221
protected application environment 35
protecting files from being accessed 62
PSERV command
See CMSPROG
PSW operand of DISPLAY command
See DIAG
PUNCH command
example 125
punch files produced by job running in batch virtual machine 212
PUNCHC macro
See CMSPROG
punching
CMS files 125
jobs to batch virtual machine 206
PURGE command 317
PURGE command to delete spool files 123
purging batch jobs 213
PUT SCREEN command 305, 322
PUT VSCREEN command 305, 322

Q

QSAM access method
See CMSPROG
QUERY
command (CMS)
display search order of disks 58
how much space is on a disk 65
query CMSPF keys 182
command (CP)
display color and extended highlight values 19
query status of CP SET MSG function 4
QUERY command 317
QUERY CPLANG command 5
QUERY LANGLIST command 5
QUERY LANGUAGE command 5
QUIT subcommand to terminate an edit session 109
QVM command 318

R

RDTERM macro
See CMSPROG
read-only extensions, using 59
read/write
status of disks
displaying 26
in VM/SP directory entry 25
read, to virtual console, definition 39
READCARD command
restriction in CMS batch facility 212
used to assign filemode numbers 62
READER operand
of ASSGN command, restriction in job for CMS batch facility 212
of FILEDEF command, restriction in job for CMS batch facility 212
reader, holding user files in 120
reading
cards from your virtual card reader 124
real card decks into your virtual machine 124
READY command 318
ready message
controlling how it is displayed 19
not displayed after #CP function is used in CMS 34
RECFM option of FILEDEF command
See CMSPROG
record length of CMS file 74
REFRESH command 306, 322
RELATED HELP 95
RELEASE command
updating master file directory 67
releasing
disks 27
read-only extensions 59
Remote Spooling Communications Subsystem (RSCS) networking 82
remote terminals, using an editor 117
RENAME command
changing filemode numbers only 73
renaming CMS files 73
renaming , CMS files 73
reordering batch jobs 213
REP operand of MACLIB command
See CMSPROG
REPEAT command 318
REPLACE statement
See CMSPROG

These symbols are used in the index to refer to other VM and VM/SP books:
CMSPROG—VM/SP CMS for System Programming, SC24-5286
DIAG—VM Diagnosis Guide, LY24-5241

REPRO access method services function
    See CMSPROG
REQUEST command 318
RESET command 318
responses
    from CMS commands 19
    from VM/SP 9
restarting batch jobs 213
RESTORE WINDOW command 254, 306, 322
restrictions
    on commands used in CMS batch facility 211
resume
    after an attention interruption 40
    program execution
        after regaining control following a disconnect 10
    terminal displays 41
RETRIEVE function for display terminals 11
retrieving previously entered data 11
RETURN CMS subset command to leave subset 36
return code in CMS ready message 19
REWIND command 318
ROUTE command 306, 322
RSERV command
    See CMSPROG
RT (Resume Type) Immediate command 41
RUNNING status on display screen 15

S

SAM (sequential access method) files
    See CMSPROG
SAVE subcommand
    changing file identifier 111
    writing file onto disk 109
SAVESYS command 318
screen
    example of 3270 screen display 30
    full-screen status notices
        enter a command or press a PA or PF key 17
        enter your response in vscreen vname 17
        executing a command 17
        scroll forward for more information in vscreen vname 17
    VM status notices
        CP READ 14
        HOLDING 16
        MORE... 16
        NOT ACCEPTED 17
        RUNNING 15
        VM READ 14
SCREEN command 318
SCREEN/NOSCREEN option of DETAIL HELP 96
SCRIPT
    files 57
    filetype usage in CMS 57
SCROLL command 306, 322
scrollable data area in full-screen CMS 176
scrolling forward and backward using Border commands 194
scrolling right and left using Border commands 197
SDATE synonym to sort FILELIST 71
search order
    for CMS commands
        displaying 65
        summary 67
    for CMS disks 58
    for filemode extensions 58
searching
    disks for CMS files
        See disk determination
    read-only extensions 58
SEND command 318
sending
    files to other virtual machine users
        from SENDFILE menu 165
        using DISK DUMP command 125
        using SENDFILE command 165
    messages to other virtual machine users
        using CP MESSAGE command 161
        using TELL command 160
    notes to other virtual machine users
        using SENDFILE command 164
SEQUENCE statement
    See CMSPROG
SET ABBREV command 76
SET BORDER command 255
SET command 319
    CONCEAL option 35
    description 319
SET command (CMS)
    controlling ready message display 19
    controlling whether you receive messages 4
    invalid forms in job for CMS batch facility 212
    operands invalid in job for CMS batch facility 212
    set tracing on or off for System Product Interpreter EXECs 234
    setting full-screen CMS 175
    setting implied CP function 34
    setting implied EXEC function 226
    setting program function keys 12
    setting the reserved line in full-screen CMS 257
    setting window borders in full-screen CMS 255
SET INSTSEG
    description 234
    use of 230, 238
SET LANGUAGE command 5
SET LOGFILE command 163, 191
SET RESERVED command 257
SET TRANSLATE command 76
SETSSI, OS linkage editor control statement
    See CMSPROG
setting
    defaults for SENDFILE command, example 165
    length of ready message 19

limits on system resources during batch jobs 209
program function keys 12, 184
screen colors and highlighting features 18
sharing
virtual disks 25
SHOW WINDOW command 306, 323
SHUTDOWN command 319
signing onto VM/SP 6
simulated data sets
filemode number of 4 63
SIZE WINDOW command 252, 306, 323
SLEEP command 319
using on display terminals 16
SLREC synonym to sort FILELIST 71
SMODE synonym to sort FILELIST 71
SMSG command 319
SMSG command (CP) 161
SNAME synonym to sort FILELIST 71
SNTMAP command 308
SORT command to specify filemode numbers 49
sorting
CMS disk files 20
files in FILELIST 71
SPACE command 319
spacing between lines of text 290
special characters
in filenames and filetypes 47
using to determine if APL is on 44
3270 Text feature 44
special messages, controlling whether you receive them 161
specifying filemode numbers on DLBL and FILEDEF commands 64
SPGEN command 308
splitting CMS files into smaller files 116
SPLOAD command 308
SPMODE command 319
SPOOL command 319
changing characteristics of unit record devices 119
spooling console output 28
used to combine multiple spool files 121
spool files
controlling in job for CMS batch facility 212
determining status of 119
produced by CMS batch facility, controlling 212
spooling
basic description 119
console output 28
multiple copies 121
SPTAPE command 319
SRECF synonym to sort FILELIST 71
SSERV command
See CMSPROG
SSIZE synonym to sort FILELIST 71
stacking
null lines
after attention interruption 41
at your terminal 3
standard label processing, CMS/DOS 141
standard labels, OS simulation 131
START command 319
STATE command
See CMSPROG
status area in full-screen CMS 177
status information in full-screen CMS 177
status notices in full-screen CMS
enter a command or press a PA or PF key 17
enter your response in vscreen vname 17
executing a command 17
scroll forward for more information in vscreen vname 17
STCP command 319
STORE command 320
See also DIAG
STYPE synonym to sort FILELIST 71
submitting jobs to CMS batch facility 205
non-CMS users 217
subsetting options in DETAIL HELP
ALL 94
DESCRIPT 94
ERRORS 94
FORMAT 94
NOTES 94
OPTIONS 94
PARMS 94
summary
of CMS commands 299
of CMS commands for system programmers 308
of CP command privilege classes 311
of CP commands 313
of HELP and HELPCONV format words 282
of Immediate commands 300
suppressing
messages sent by the TELL command 161
special messages 161
suppression of passwords on the command line 25
SVCTRACE command
See DIAG
symbols
logical line editing 295
SYNONYM
command to invoke synonym tables 75
filetype usage in CMS 54
used to define synonyms for CMS and user-written commands 75
synonyms used to sort FILELIST
SDATE 71
SLREC 71
SMODE 71
SNAME 71
SRECF 71
SSIZE 71
STYPE 71
SYSPROF EXEC 13
SYSTEM command 320

These symbols are used in the index to refer to other VM and VM/SP books:
CMSPROG—VM/SP CMS for System Programming, SC24-5286
DIAG—VM Diagnosis Guide, LY24-5241

system disk, files available 63
System Product Editor
  considerations for migrating to full-screen CMS 333
  example, of using 112
  full screen display 112
  invoking 110
  line-mode on display terminals 117
  using to display HELP files 104
System Product Interpreter
  basic description 221
  invoking 226
  REXX language, interpreted by 221
  sample EXECs 227
  writing EXECs for the 225
system profile 13, 79
SYSUTx filetype 56
SYSUT1 filetype 56
SYSUT2 filetype 56
SYSUT3 filetype 56
SYSUT4 filetype 56
SYS00x filetype 56
SYS001 filetype 56
SYS002 filetype 56
SYS003 filetype 56
SYS004 filetype 56
SYS005 filetype 56
SYS006 filetype 56

T

TAG command 320
TAPE command
  creating CMS files from tapes created by 126
  sample terminal display 128
  using 127
tape file
  DCB address 136
  FCBSECT address 136
tape files, in CMS 126
tape handling options, specifying 151
tape label
  by CMS commands 143
  EOT 146
  EOV 146
  LABELDEF command 145
  MOVEFILE command 150
  under CMS/DOS 140
    DTFMT macro 140
  under OS simulation 131
  under OS/VS simulation 137
tape label, processing, IBM standard 131
tape labels
  in CMS 130
  limitations 130
TAPECTL macro used in tape label process 143
TAPEMAC command 143
tapes
  bypass label, description 134
  compacting output 127
  density, when to specify 151
  label processing 131
  labels
    in CMS 130
    in CMS/DOS 140
    in OS simulation 131
  nonlabeled, description 134
  nonstandard label, description 135
  optional handling 151
  special handling 151
  virtual addresses 126
TAPESL macro, description 143
TAPPDS command 143
  copying files from tapes 150
  creating CMS files from tapes created by 127
TASK menu 99
TASK menus 274
TCLOSE command in tape label processing 139
TE (Trace End) Immediate command 234
TERMINAL command 320
TERMINAL command (CP) used to set logical line editing symbols 297
terminals
  characteristics 14
  disconnecting 10
  display
    See display terminals
  mode, setting 39
  requesting information about 81
terminating
  editing session 22
  execution of a program 40
  terminal session 10
TEXT
  files
    created by assembler language processors 55
  filetype
    usage in CMS 54
text feature for 3270 terminals 44
title line in full-screen CMS 176
TO operand of SPOOL command 122
toggling between pf keys 101
TRACE command 320
  See also DIAG
tracing program activity
  See DIAG
TRANSFER command 320
  description 320
  moving reader files 123
transferring
  data files 123
transient area
  CMS commands that execute in 68
translating output characters 290
translation synonyms for commands 76
translations for commands 76
TRUNC option of COPYFILE command 74

truncating trailing blanks 74
TS (Trace Start) Immediate command 234
TSOMAC MACLIB
  See CMSPROG
TXTLIB command
  See CMSPROG
TXTLIB filetype usage in CMS 54
type call in tape label processing 137
TYPE command 23
TYPE/NOTYPE option of DETAIL HELP 96
TYPEWRITER subcommand used to edit in line-mode 118

U

underscore
  highlighting feature, controlling with CP SCREEN command 18
  in filename and filetype 48
unit record, devices 119
unlabeled tapes, defining 140
UNLOCK command 320
UPDATE filetype usage in CMS 54
updating
  CMS file directories 66
UPDLOG filetype usage in CMS 54
UPDTxxxx filetype usage in CMS 55
UPSI operand of CMS SET command
  See CMSPROG
user file directory 66
user hold status of spool files 120
user program area
  commands that execute in 68
user-written
  EXECs, samples 227
userid
  for CMS batch virtual machine 205
  for your virtual machine 6
  specifying for output spool files 122
using XEDIT subcommand in HELP 104
UTILITY command 309

V

VALIDATE command 307
VARY command 320
vector
  cleanup 224
  resetting using EXECOS 238
virtual addresses
  for disks 21
  for tapes 126
  for unit record devices 119
virtual disks
  See also DISK command
  definition 23
Virtual Machine/System Product
  See VM/SP (Virtual Machine/System Product)
Virtual Machine/System Product (VM/SP)
  basic description 3
  command summaries 299
  components of 3
  environments 31
virtual machines
  definition 3
virtual screen
  default characteristics 264, 324
  defining 248
  general description 173
  system-defined 179
virtual storage
  requesting information about 81
  used by editor, what to do when it is full 114
VM READ status on display screen 14
VM/SP (Virtual Machine/System Product)
  basic description 3
  command summaries 299
  components of 3
  environments 31
VM/SP System Product Editor
  See System Product Editor
VM/SP System Product Interpreter
  See System Product Interpreter
VMDUMP command 320
VMFASM command 309
VMFASM EXEC procedure
  See CMSPROG
VMFDOS command
  See CMSPROG
VMFLKED command 309
VMFMAC command 309
VMFMERGE command 309
VMFNLS command 309
VMFPLC2 command 309
VMFREMOV command 309
VMFTXT command 309
VMFZAP command 309
VMSERV command 309
VOLID parameter FILEDEF command 132
VRSIZE command 309
VSAM cleanup 224
VSAM option of DLBL command
  See CMSPROG
VSAM, CMS support
  See CMSPROG
VSAPL program, invoking 42
VSE
  differences between CMS/DOS tape label processing 141
  TLBL card in tape label processing 140
VSEVSAM command
  See CMSPROG

These symbols are used in the index to refer to other VM and VM/SP books:
CMSPROG—VM/SP CMS for System Programming, SC24-5286
DIAG—VM Diagnosis Guide, LY24-5241

W

WAITREAD VSCREEN command 234, 242, 307, 323
WAITT macro
  See CMSPROG
WAITT VSCREEN command 307, 323
WARNING command 320
WARNING LOGFILE 191
window
  characteristics of 175
  default characteristics 262
  defining 248
  dropping 187
  general description 173
  maximizing 253
  popping 187
  positioning 251
  restoring 254
  setting borders 255
  sizing 252
  system-defined 179
windowing and full-screen CMS considerations 321
WM window
  customizing 247
  messages 180
  PF keys
    Backward 186, 329
    Clear 186, 329
    Copy 186, 329
    Forward 186, 329
    Help 186, 329
    Left 186, 329
    Maximize 186, 329
    Quit 186, 329
    Restore 186, 329
    Retrieve 186, 329
    Right 186, 329
    Top 186, 329
  using 180, 198
WRITE VSCREEN command 249, 307, 323
writing
  applications in full-screen CMS 334
  CMS files onto disk, disk determination 61
  labels on CMS disks 24
WRTERM macro
  See CMSPROG
WVOL1 function in tape command processing 144

X

XEDIT
  CLOCATE subcommand 105
  command to invoke System Product Editor 110
  example 112
  subcommands, invoking 5
XMITMSG command 307

Z

ZAP filetype usage in CMS 55

Numerics

19E virtual disk address accessed as Y-disk 58
190 virtual disk address accessed as S-disk 58
191 virtual disk address accessed as A-disk 48
192 virtual disk address accessed as D-disk 58
3270 screen display, example 30
3270 terminals
  See display terminals

VM/SP CMS User's Guide
Order No. SC19-6210-4

READER'S COMMENT FORM

Is there anything you especially like or dislike about this book? Feel free to comment on specific errors or omissions, accuracy, organization, or completeness of this book.

If you use this form to comment on the online HELP facility, please copy the top line of the HELP screen.

____________ ____________ _______ Help Information line ____ of ____

IBM may use or distribute whatever information you supply in any way it believes appropriate without incurring any obligation to you, and all such information will be considered nonconfidential.

Note: Do not use this form to report system problems or to request copies of publications. Instead, contact your IBM representative or the IBM branch office serving you.

Would you like a reply? ___YES ___NO

Please print your name, company name, and address:

____

____

____

____

IBM Branch Office serving you: ____________________________________

Thank you for your cooperation. You can either mail this form directly to us or give this form to an IBM representative who will forward it to us.

SC19-6210-4

Reader's Comment Form

CUT
OR
FOLD
ALONG
LINE

Fold and tape   Please Do Not Staple   Fold and tape

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL
FIRST-CLASS MAIL   PERMIT NO. 40   ARMONK, NY

POSTAGE WILL BE PAID BY ADDRESSEE:

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION
DEPARTMENT G60
PO BOX 6
ENDICOTT NY 13760-9987

Fold and tape   Please Do Not Staple   Fold and tape

IBM®

International Business
Machines Corporation
P.O. Box 6
Endicott, New York 13760

File No. S370/4300-39
Printed in U.S.A.

SC19-6210-4

International Business
Machines Corporation
P.O. Box 6
Endicott, New York 13760

File No. S370/4300-39
Printed in U.S.A.

SC19-6210-4

SC19-6210-04