VM/Pass-Through Facility

SC24-5557-01

Administration and Operation

Version 2

Note!

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

This book is also provided as an online book that can be viewed with the IBM BookManager* READ and IBM Library Reader* licensed programs.

Second Edition (June 1993)

This edition, SC24-5557-01, applies to the VM/Pass-Through Facility, Version 2, Release 1.0 and Release 1.1, program number 5684-100, and to all subsequent releases of this product until otherwise indicated in new editions or Technical Newsletters. Changes are made periodically to the information herein.

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

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 person or company.

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 reader's 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, 1701 North Street, Endicott, NY, U.S.A. 13760-5553. IBM may use or distribute whatever information you supply in any way it believes appropriate without incurring any obligation to you.

© Copyright International Business Machines Corporation 1991,1993. All rights reserved.
Note to U.S. Government Users — Documentation related to restricted rights — Use, duplication or disclosure is subject to restrictions set forth in GSA ADP Schedule Contract with IBM Corp.

Contents

Notices xv
Programming Interface Information xv
Trademarks and Service Marks xvii

Preface xix
What this Book Contains xix
How to Use this Book xx
Sending Comments for this Publication xxi

Summary of Changes xxiii
Changes for PVM Version 2: xxiii
Additions for PVM Version 2 Release 1.1: xxiii

Chapter 1. Introduction 1
What is PVM? 1
How PVM Enhances Your VM System 2
Networking Terms 4
What You Can Access Through PVM 5
How You Can Access PVM 7

Chapter 2. CVIEW - Product Overview 9
What is CVIEW? 9
Modes of Operation 10
Setup Mode 10
Session Mode 11
Notify Mode 12
Control Mode 13
Technical Overview 13
Accessing the CVIEW Virtual Machine 13
Networking Sessions 14
Networking Considerations 15
PASSTHRU Considerations 16
Display-input Option 16
Accommodating Different Display Station Types 17
Messages 17

Part 1. Administering, Installing, and Operating PVM 19

Chapter 3. Administrative Considerations 21
What's New In PVM Version 2 21
Installation Requirements 22
Device Support 23
Storage Considerations 25
DASD Requirements 25
Storage Needs in the Default Configuration 26
Modifying Your Storage Needs 27
How Much Storage Is Needed in Each User's Virtual Machine for MPVM? 27
How Much Storage Is Needed for Your PVM Server? 28
How Much Storage is Need for Your PVMG Server? 29
Performance Considerations 29
Security Considerations 29

© Copyright IBM Corp. 1991,1993

Notepad Facility . . . 29
Session Security . . . 30
Restrictions . . . 32
General Restrictions . . . 32
Access Security Exits Restrictions . . . 34
PCCF Restrictions . . . 34
Channel to Channel Adapter (CTCA) Restrictions . . . 34
3271/3274 Emulator Restrictions . . . 34
Logical Device Support Restrictions . . . 35
3270 Information Display System Printer Restrictions . . . 35
ROCF Restrictions . . . 36
Migrating from PVM 1.4 . . . 36
Using PVM Version 2 in a VM/ESA or VM/XA Environment . . . 37
Virtual Machine Mode . . . 37
Line Drivers . . . 37
PVM TRACE Command . . . 37
Virtual Machine Dumps . . . 37
CP Remote Support . . . 37
Hardware . . . 38
Performance . . . 38
Requirements . . . 38
Installing PVM 2.1 on a VM/ESA or VM/XA System . . . 38
Storage Considerations . . . 39
ESCON Channel Support . . . 39

Chapter 4. Installing PVM 2.1 . . . 41
Step 1. Add Directory Entries for the PVM and PVMG Operators . . . 41
Sample PVM Directory . . . 42
Sample PVMG Directory . . . 43
Step 2. The MAINT (or Installer's) User ID . . . 44
Step 3. Getting Disks for PVM, PVMG, and MPVM Files . . . 44
Step 4. Load the Tape Files to Disks . . . 46
Loading Files - The Quick Method (Using INSTFPP) . . . 48
Loading Files - The Standalone Method . . . 48
When to Install the National Language Feature . . . 50
Step 5. Set Up Autolog Statements for PVM and PVMG . . . 51
Step 6. Set Up the PVMG Server Virtual Machine . . . 52
Step 7. Add PVMG to GCS . . . 52
Step 8. Add VTAM Definitions . . . 53
Step 9. Apply Service . . . 54
Step 10. Verify Initial Installation . . . 54

Chapter 5. CVIEW Installation Planning . . . 57
CVIEW Virtual Machine Requirements . . . 57
Disk Space . . . 57
Virtual Storage . . . 57
Privilege Class . . . 57
Channel Mode and Nonshared Virtual Control Units . . . 57
User ID . . . 57
CVIEW Directory Entry . . . 58
CVIEW Operating Environment . . . 58
Machine Requirements . . . 58
Multiple CVIEW Virtual Machines . . . 59

Chapter 6. Defining Your PVM Configuration . . . 63

Planning a Network 63
Creating the Configuration File 66
Sample Configuration Files 66
Commenting the PVM Configuration File 70
Identifying the System Language 70
Excluding Double-Byte Character Messages 71
Identifying the Local Node Name 71
Identifying a Unique Resource Name 71
Defining a Link from a Local Node to Another Node 71
Adding Devices to a Line 72
Defining Groups 73
Defining Routes and Alternate Routes 74
Authorizing the PVM Operator 75
Identifying CMS Subset Mode Commands to PVM 75
Automatically Connecting Users to a Node 76
Setting PF Keys for PVM Menus 76
Saving PVM Information 77
Specifying Restrictions 77
Specifying Message Requirements 78
Specifying Storage Requirements 79
Creating a CMS PROFILE EXEC for PVM 79
Creating a CMS Communications Directory 80
Creating a PVM Profile Exec 81

Chapter 7. Defining Your PVMG Configuration 85
Planning Your PVMG Configuration 85
Planning the PVMG Server's Role in the PVM Network 86
Commenting the PVMG Configuration File 89
Identifying the System Language 89
Excluding Double-Byte Character Messages 89
Identifying the Local Node Name and the VTAM Application Name 90
Defining a Link from PVMG to PVM 90
Defining the VTAM APPLID to Which Users Log on 90
Assigning Node Names to PF Keys 90
Defining Routes and Alternate Routes 90
Specifying LU Names Used by PVMG with Secure VTAM APPLIDs 91
Authorizing PVMG Operators 91
Saving PVMG Information 91
Specifying Message Requirements 91
Specifying the Target User ID for Natural Language Message Reception . 92
Specifying the User ID of the Programmed Operator Virtual Machine That Receives Encoded Messages 92
Specifying the VTAM APPLID to Disconnect When PVMG Fails 92
Changing the Default Local PVMG Node Session Termination String 92
Specifying VTAM Logon Model Table Entries Used by PVMG 92
Tailoring the PROFILE GCS File 93

Chapter 8. Developing NAMES Files for MPVM 95
Creating Entries for Applications 95
Specifying the Nickname 96
Specifying Characteristics for Application Nicknames 96
Defining Target Applications 96
Defining Other Variables for Applications 97
Specifying Commands to be Invoked 97
Specifying Your Own Subtags 97

Specifying Nickname Lists . . . 97
Summary of Application NAMES File Entries . . . 97
Defining Configuration Settings—:nick.?CONFIG . . . 98
User Initialization Settings . . . 98
Creating Other Special Entries . . . 98
Setting Global Values for All Sessions—:nick.?INITIAL . . . 98
Setting Default Values for All Sessions—:nick.?DEFAULT . . . 98
Cleaning Up After the Last Session Has Ended—:nick.?FINAL . . . 99
Sample MPVMSYS NAMES File . . . 99
Verifying the MPVMSYS NAMES File . . . 100
Considerations for Individual User NAMES Files . . . 100
Summary of MPVMSYS NAMES Contents . . . 101

Chapter 9. Operating the Network . . . 103
Starting PVM . . . 103
Querying the System . . . 107
Defining a Link Dynamically . . . 109
Displaying the Path of a Session . . . 110
Sending Messages . . . 111
Issuing CMS Subset Mode Commands . . . 111
Quiescing, Resuming, and Dropping Links, Devices, and Users . . . 112
Using the PVM Programmable Operator (PVMPROP)/Command Response Interface (CRI) . . . 114
Stopping PVM . . . 114
Starting and Stopping Printer Sessions . . . 115
Printer Session Initiation . . . 115
Printer Session Termination . . . 117
Starting PVMG . . . 117
Querying PVMG . . . 117
Querying the System . . . 117
Monitoring User Activity . . . 118
Displaying the Path of a PVMG Session . . . 118
Sending Messages . . . 118
Dropping Users and Nodes . . . 119
Stopping PVMG . . . 120

Chapter 10. Using CVIEW . . . 123
Starting the CVIEW Virtual Machine . . . 123
Automatically Starting CVIEW . . . 123
Manually Starting CVIEW . . . 123
CVIEW Commands . . . 124
CMS . . . 124
CP . . . 125
DROP . . . 125
HELP . . . 126
PASSWORD . . . 126
QUERY . . . 127
SHUTDOWN . . . 129
STOP . . . 130
Establishing a Session . . . 130
Managing a CVIEW Session . . . 133
Consult Sessions . . . 133
Conference Sessions . . . 134
Device Type and Model Assignment . . . 136
Joining a Conference Session in Progress . . . 136

CVIEW PROFILE EXEC . . . 137
CVIEW Log File . . . 137

Part 2. Servicing and Determining Problems . . . 139

Chapter 11. Applying Service to PVM 2.1 . . . 141
Updating National Language Support Files . . . 141
Applying Service Using VMSES/E . . . 141
Applying Service without VMSES/E . . . 141
Applying Preventative Service (PUT Tape) . . . 142
Applying Preventative Service Using VMSERV . . . 143
Running the Service Exec Without VMSERV . . . 144
Applying Preventative Service Standalone . . . 144
Applying Corrective Service . . . 147
Manually Applying Corrective Service for MODULE and LOADLIB Users 147
Manually Applying Corrective Service for TEXT Users . . . 152
Manually Applying Corrective Service for SOURCE Users . . . 160
Automatically Applying Corrective Service . . . 168
CORRSERV EXEC Package . . . 168
5684100 HISTORY File . . . 169
Preparing To Run CORRSERV . . . 172
Understanding How CORRSERV Processes . . . 173
PLAN Stage . . . 173
APPLY Stage . . . 174
CORRSERV . . . 175
Location of PVM's Files on MAINT's Minidisks . . . 175

Chapter 12. Collecting Problem Information . . . 179
Trace Facility for the PVM Component . . . 180
System Services Tracing . . . 180
TRACE USER Command . . . 180
Tracing Line Driver I/O Activity . . . 181
Line Driver Data Transfer Tracing . . . 182
PCCF Tracing . . . 182
Tracing System Activity . . . 182
The Dump Facility . . . 187
Full Dumps . . . 188
Partial Dumps . . . 188
Analyzing Dump Information . . . 189
Creating A Problem Report . . . 189
Examining the Dump . . . 190
Other PVM Problem Determination Actions . . . 190
Tracing MPVM . . . 190
TRACEn Subcommand . . . 192
Logging Messages and Screen Images . . . 193
Diagnosing PVMG Problems . . . 193
PVMG Trace Facility . . . 193
Tracing PVMG System Activity . . . 194
PVMG I/O Tracing . . . 194
Other PVMG Problem Determination Actions . . . 194
The PVMG Dump Facility . . . 194
Analyzing PVMG Dump Information . . . 195

Part 3. Modifying PVM 197

Chapter 13. Tuning PVM Performance 199
Enhancing PVM Performance 199
SET QDROP OFF Option 199
SET FAVORED Option 199
VCUNOSHR Directory Option 199
DIAGNOSE X'98' Real Channel Program Directory Option 199
SET RESERVE Command 200
Dispatching Priority 201
ROCF Sessions 201
Tailoring the PVM Module 201
Installing MPVM in a Shared Segment 203
MPVM Shared Segment Verification 204
Changing the Names of MPVM Shared Segments 204
Sample DMKSNT Entries for VM/SP or VM/ESA 370 based Systems . . . 205
Adding Common MPVM Commands and Macros to Your Installation Shared Segment 205
Saving MPVM Common Disk Information in a Segment 206
Fine-Tuning the PVM Server 207
Checking Response Time for User Commands 207
Customizing MPVM Sessions 207
Sample MPVM Exits 208
Developing Exits 209
Programming Note 210
PVM Macros Used in the Sample Access Security Exits 210
DVMENTR 211
DVMRETN 211
DVMXTRN 212
Session Control Exits 212
Session Initiation Exit (DVMNUE1) 212
Session Termination Exit (DVMNUE2) 213
Session Request Exits (DVMNUE3 and DVMNUE4) 214
Command and Response Exit (DVMNUE5) 215
CINIT Session Initiation Data Exit (DVMNUE6) 216
Sample Session Control Exit 217
Network Management Exits 218
Cluster Statistics Exit (DVMNMEC) 219
Link Statistics Exit (DVMNMEL) 219
Session Statistics Exit (DVMNMES) 221
CMS User Exits 221
Before Initialization Exit (DVMCUEBI) 222
After Initialization Exit (DVMCUEAI) 222
Before Read Exit (DVMCUEBR) 223
After Read Exit (DVMCUEAR) 223
Before Write Exit (DVMCUEBW) 224
After Write Exit (DVMCUEAW) 224
Before Final Exit (DVMCUEBF) 225
Rebuilding PVM for Exits 225

Chapter 14. National Languages on Your PVM System 227
National Language Overview 227
Languages Available Through IBM 227
Backup National Language 228

Installing National Languages for PVM 2.1 . . . 228
Step 1. Getting Disks for National Language Files . . . 228
Step 2. Loading Files From Tape to Disk . . . 230
Step 3. Enabling PVM Components for Additional Languages . . . 233
Step 4. Verifying National Language Installation . . . 233
For PVM . . . 233
For MPVM . . . 234
For PVMG . . . 234
Changing the Backup Language . . . 235
Changing Message Destinations . . . 236
Changing the PVM Translate Table . . . 237
Creating and Using a Message Repository . . . 238
Creating a Message File . . . 238
Compiling the Message File . . . 238
Verifying the Message File . . . 239
Making the Message File Available . . . 239
Rules for Making Your Own Repository . . . 240
Comment Record . . . 240
Master Record . . . 241
Time Stamp Records . . . 242
Message and Screen Records . . . 243
Substitution Variables in Messages . . . 245
Special Considerations for Message and Screen Records . . . 246
Control Lines . . . 247
Message Repository Samples . . . 248

Chapter 15. Facilities for Automating PVM and PVMG Operations . . . 251
PVMPROP - A Programmable Operator Interface for PVM and PVMG . . . 251
The Command Response Interface . . . 252
PVMPROP/CRI Message Protocol . . . 252
Message Format . . . 253
Deciphering Encoded Messages . . . 253
Using Command Prefixes . . . 255
Sample Application . . . 257

Chapter 16. Application-to-Application Overview . . . 259
PCCF Data Flow . . . 259
Data Transmission . . . 260
Session Termination . . . 260
PCCF Application-to-Application Restrictions . . . 261
Application-to-Application Communication Sample Program . . . 261

Chapter 17. Customizing CVIEW . . . 265
Listed Consultants — CONSULT IDNAME . . . 265
Defining Listed Consultants . . . 265
Program Function Key Assignments — DXGKEYS LIST . . . 266
PF Key Functions . . . 266
Changing PF Key Assignments . . . 266
The Customizing EXEC — DXGINIT . . . 267
Concurrent Users . . . 267
The NOTIFY Panel . . . 267
Privileged Users . . . 268
Executing the Customization EXEC . . . 268

Part 4. Reference Information 271

Appendix A. PVM Macros Supported as Programming Interfaces 275
Product-Sensitive Programming Interface Executable Macros 275

Appendix B. Tape Layouts for PVM 2.1 277
Distribution Tape 277
Service Tape 278
National Language Feature Tape 279

Appendix C. PVM Commands 281
How to Read the Syntax Diagrams 281
* (Comment) 286
AUDIT 286
AUTHORIZ 287
CDBC 289
CMD 290
CMS 290
CP 291
CPQ 292
DEFINE 293
BSCA Format 294
CTCA Format 294
APPC Format 294
IUCV Format 295
3088 Format 295
DISC 298
DROP 299
EXEC 301
LMSG 302
LOGMSG 303
MPVM 304
MPVMBLD (PVM 2.1) 307
MSG 308
PASSTHRU 309
PAUSE 313
PMSG 314
PVMBLD 315
PVMCMP 316
PVMGEN (PVM 2.1) 317
QUERY 318
QUIESCE 321
QUIT 323
RESUME 323
ROUTE 324
RUNPVM 325
SELECT 326
SET 326
SHOW 328
SHUTDOWN 329
START 330
STATUS 332
TRACE 334
VARY 337

CP Commands Used with PVM 338
DIAL 338
SMSG 339
PVM/IPCS Support Commands 339
COMBLOCK 340
DEVQUE 340
HELP 341
IPCSDUMP 341
IPCSSCAN 342
LINKS 343
MAP 344
ROUTES 345
SYSBLOCK 345
TASKQUE 346
USERQUE 346

Appendix D. PVM Configuration File Records 347
* (Comment) 347
AUDIT 347
AUTHORIZ 349
AUTO 349
BUFFERS 350
CLPORT 351
CLUSTER 353
DUMP 353
EXT 354
GROUP 355
I/O 356
LANG 357
LINK 358
LNODE 360
LOCAL 361
MAXCONN 362
MAXIDLE 362
MAXSESS 363
MAXSUSP 363
MAXTDISC 364
MAXUSERS 364
MSGNOH 365
MSGOPTNS 365
OPID 366
PORT 367
PVMPROP 369
RESID 370
ROUTE 370
SELECT 371
SESSTERM 371
STATTIME 372
SUBSET 372
TDISC 373
TIMEOUT 373
XDBC 374

Appendix E. PVMG Commands 375
How to Read the Syntax Diagrams 376

AUTHORIZ . . . 378
CDBC . . . 380
CMD . . . 381
DISC . . . 382
DROP . . . 382
INIT . . . 383
LMSG . . . 384
LOGMSG . . . 385
MSG . . . 385
PMSG . . . 386
PVMGBLD (PVM 2.1) . . . 387
QUERY . . . 388
QUIESCE . . . 390
RESUME . . . 391
ROUTE . . . 391
RUNPVMG . . . 392
SELECT . . . 393
SHOW . . . 393
SHUTDOWN . . . 394
SPOOL (PVM 2.1.1) . . . 396
START . . . 397
TRACE . . . 398
CP Command Used with PVMG . . . 399
SMSG . . . 399

Appendix F. PVMG Configuration File Records . . . 401
* . . . 401
APPLN . . . 402
AUTHORIZ . . . 403
DUMP . . . 403
LANG . . . 404
LINK . . . 404
LNODE . . . 405
LOCAL . . . 406
LOGMODE . . . 406
LOGONID . . . 407
LU . . . 408
MSGNOH . . . 410
MSGOPTNS . . . 410
OPID . . . 411
PVMPROP . . . 412
ROUTE . . . 412
SELECT . . . 413
SESSDISC . . . 413
SESSTERM . . . 414
XDBC . . . 414

Appendix G. Setting Up an APPC Link . . . 417

Appendix H. PVM Control Block Formats . . . 419

Appendix I. Manually Dialing a Remote 4300 Processor . . . 437

Appendix J. PVM Access Security Exits . . . 439
Installing and Building the Security Exits for PVM 2.1 . . . 440

The REMOTE AUTH File 440
Using Generics 441
Reloading the REMOTE AUTH File 441
Definitions of the REMOTE AUTH File Entries 442
GATELINK 442
REM-CMD 443
REM+USER 444
SNA-USER 445
VM-USER 446
Messages 448
Messages Returned to the User: 448
Messages Returned to the PVM Console 448
Format of SECAUDIT LOG Records 449
Controlling Local Users Using the REMOTE AUTH Table 450
DVMNUE3 Testing 451
SECAUDIT LOG 452
Controlling Remote Users by Using the REMOTE AUTH Table 453
DVMNUE4 Testing 454
SECAUDIT LOG 455

Appendix K. CVIEW Product Panels 457
Setup-mode Panels 457
Consult Session Setup (Recipient) 458
Consult Session Setup (Consultant) 460
Conference Session Setup (Attendee) 462
Conference Session Setup (Moderator) 464
Notify-mode Panels 465
Control-mode Panels 467
Consultant Control 467
Moderator Control 468

Glossary 471

Bibliography 509
VM/Pass-Through Facility Library 509
RSCS Library 509
VM/ESA Release 2.1 Library 510
Related Books 511

Index 513

Notices

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

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

For online versions of this book, we authorize you to:

- Copy, modify, and print the documentation contained on the media, for use within your enterprise, provided you reproduce the copyright notice, all warning statements, and other required statements on each copy or partial copy.
- Transfer the original unaltered copy of the documentation when you transfer the related IBM product (which may be either machines you own, or programs, if the program's license terms permit a transfer). You must, at the same time, destroy all other copies of the documentation.

You are responsible for payment of any taxes, including personal property taxes, resulting from this authorization.

THERE ARE NO WARRANTIES, EXPRESS OR IMPLIED, INCLUDING THE WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

Some jurisdictions do not allow the exclusion of implied warranties, so the above exclusion may not apply to you.

Your failure to comply with the terms above terminates this authorization. Upon termination, you must destroy your machine readable documentation.

Programming Interface Information

This administration and operation guide is intended to help you install and maintain the VM/Pass-Through Facility (PVM).

This administration and operation guide also documents General-Use Programming Interface and Associated Guidance Information, Product-Sensitive Programming Interface and Associated Guidance Information, and Diagnosis, Modification, and Tuning Information provided by VM/Pass-Through Facility (PVM).

General-Use programming interfaces allow the customer to write programs that obtain the services of the VM/Pass-Through Facility.

© Copyright IBM Corp. 1991,1993

General-Use Programming Interface and Associated Guidance Information is identified where it occurs, either by an introductory statement to a chapter or section or by the following marking:

GUPI

General-Use Programming Interface and Associated Guidance Information...

GUPI end

Product-Sensitive programming interfaces allow the customer installation to perform tasks such as diagnosing, modifying, monitoring, repairing, tailoring, or tuning of VM/Pass-Through Facility. Use of such interfaces creates dependencies on the detailed design or implementation of the IBM software product. Product-Sensitive programming interfaces should be used only for these specialized purposes. Because of their dependencies on detailed design and implementation, it is to be expected that programs written to such interfaces may need to be changed in order to run with new product releases or versions, or as a result of service.

Product-Sensitive Programming Interface and Associated Guidance Information is identified where it occurs, either by an introductory statement to a chapter or section or by the following marking:

PSPI

Product-Sensitive Programming Interface and Associated Guidance Information...

PSPI end

Diagnosis, Modification, and Tuning Information is provided to help the customer perform the diagnosis and tuning tasks of the VM/Pass-Through Facility.

WARNING: Do not use this Diagnosis, Modification, and Tuning Information as a programming interface.

Diagnosis, Modification, and Tuning Information is identified where it occurs, either by an introductory statement to a chapter or section or by the following marking:

DMTI

Diagnosis, Modification, and Tuning Information...

DMTI end

Trademarks and Service Marks

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

ACF/VTAM	
Customer Information Control System	CICS
Enterprise System Connection	ESCON
ESA/370	PC AT
BookManager	Library Reader
IBM	IBMLink
PC/XT	PS/2
PROFS	PVM/IPCS
RISC System/6000	RS/6000
System Network Architecture	SNA
Time Sharing Option	TSO
Virtual Machine/Enterprise Systems Architecture	VM/ESA
Virtual Machine/Extended Architecture	VM/XA
VTAM	VM/VCNA
Virtual Machine/VTAM Communications Network Application	

Preface

This book is for anyone that has to use or maintain VM/Pass-Through Facility (PVM) on a VM system. This book covers the information you need to install, operate, and manage PVM. In this book, the term VM refers collectively to these systems:

- Virtual Machine/System Product (VM/SP), 5664-167,
- Virtual Machine/System Product High Performance Option (VM/SP HPO), 5664-173,
- Virtual Machine/Extended Architecture System Product (VM/XA System Product), 5664-308,
- Virtual Machine/Enterprise System Architecture (VM/ESA) Version 1, Release 1.0 (5684-112) with either of the following features:
  - 370 Feature
  - ESA Feature
- VM/ESA Release 1.1 (5684-112) or later.

VM and VM/ESA are further qualified where the need exists. Qualifications for VM/ESA (ESA Feature) and VM/ESA (370 Feature) are made where necessary,

General users should refer to the *VM/Pass-Through Facility User's Guide* for information about using PVM.

In this book, VM/Pass-Through Facility Version 2 is referred to as PVM. VTAM refers to the Virtual Telecommunications Access Method Version 3 Release 2 or later. The VM/Pass-Through Facility Gateway component is referred to as PVMG, and PVM's simultaneous multiple session component, the VM/Pass-Through Facility Multiple Session Application, is referred to as MPVM.

For our purposes, a PVM system, or PVM network, can, but does not have to, include PVM's PVMG, and MPVM components. This book is arranged by groups of tasks to help make PVM easier to use.

What this Book Contains

The main parts of this book are:

Part 1, "Administering, Installing, and Operating PVM"
PVM is a program that runs on a VM system. This program defines a network that allows system users to access other systems, applications or printers. Read this part if you have to plan, install, define, or maintain a PVM system.

Part 2, "Servicing and Determining Problems"
This part helps you determine if a problem exists in your PVM system. It informs you how to start and keep PVM running so people can use the PVM functions.

Part 3, "Modifying PVM"
System programmers can customize PVM to extend the functions of PVM. This part lists parameters programmers need to be aware of in order to collect data from PVM. Refer to this part if you want to analyze network activity or to run test systems.

© Copyright IBM Corp. 1991,1993

Part 4, "Reference Information"
Once you are using or operating PVM, you may need to remind yourself of a particular item. This part includes appendix references to these topics:

- PVM macros supported as programming interfaces
- Tape layouts
- Commands and configuration file records for PVM
- Commands and configuration file records for PVMG
- Setting up an APPC link
- PVM control block formats
- Manually dialing a remote 4300 processor
- PVM sample access security exits
- CVIEW product panels

How to Use this Book

Color is used in this book to illustrate:

- Words or commands you type on your keyboard.
- Part or all of what you see on your display station's screen.

Syntax diagrams are used to describe command syntax and configuration file records. See "How to Read the Syntax Diagrams" on page 281 for instructions on reading syntax diagrams.

PVM users fit into the following categories:

- Administrator
- Operator
- System Programmer
- General user.

This book will help you better if you can place yourself, or what you are trying to do while using PVM, into one of these categories.

Administrators plan a PVM network. They assure that the general users are satisfied with the services they get from PVM. Some administrators are also responsible for installing and servicing the PVM product. Administrators may be responsible for recognizing problem conditions. Administrators should review Part 1, "Administering, Installing, and Operating PVM," and Part 2, "Servicing and Determining Problems."

Operators manage PVM and its components. Operators perform the daily tasks that keep PVM running. Operators can start PVM, enable users, and stop PVM. They can add systems or delete systems from a PVM network. Operators need to recognize problem conditions. Depending on the installation, both the operator and the administrator for PVM may be the same person. Operators need to review Part 1, "Administering, Installing, and Operating PVM."

System programmers customize PVM for specific needs of the company or department. PVM provides ways to add routines to the product to collect data or run tests. Programmers may be called in to analyze problem conditions. PVM has facilities to look at storage or trace network activity. System programmers can assist in keeping PVM available for general users and operators. Part 3, "Modifying PVM," is written for system programmers.

System programmers who wish to take advantage of the multiple session programmability of PVM's MPVM component should also consult the

VM/Pass-Through Facility Programming Reference, and the *VM/Pass-Through Facility Messages*.

General users should consult the *VM/Pass-Through Facility User's Guide* for information about general user tasks.

Sending Comments for this Publication

You can send comments using the form in the back of this publication, or use electronic mail or FAX. Please include:

- Publication title
- Order number (including suffix)
- Page, section title, or topic

The electronic addresses are:

Format	Address
INTERNET	pubrcf@gdlvme.vnet.ibm.com
IBMMAIL	USIB2L8Z@IBMMAIL
IBMLink*	GDLVME(PUBRCF)
FAX	(607)752-2327

Your comments will help improve our information. Thank you.

Summary of Changes

This section describes the changes in this book.

Changes for PVM Version 2:

- Improved MPVM shared segment instructions.
- Improved corrective service instructions.
- Added a corrective service exec (CORRSERV) for PVM 2.1 through APAR VM55703.
- CMS APAR VM34760 is required if running on a SP5/HPO5 system.
- Added PVMG LNODE configuration file record.
- Added a default option on the PVMG LU configuration file record.
- Removal of SVM from the PVM product tapes because it is now available as part of CMS. In order to use MPVM, the following APARs are required to obtain SVM:

APAR Number	Release	PTF Number
VM54782	VM/SP5 VM/HPO5	UV90721
VM54782	VM/SP6 VM/HPO6	UV90722
VM54781	VM/XA2.1	UM90121
VM54780	VM/ESA1.0	UM90118
VM54780	VM/ESA1.1	UM90119
VM54780	VM/ESA1.2	UM90120

SVM will be available as part of the CMS base for all subsequent releases of VM/ESA.

Additions for PVM Version 2 Release 1.1:

- Cooperative Viewing Facility (CVIEW) Version 2, Release 1, Modification Level 3 (5664-296) program product is being shipped with PVM as a separate component.
- Installation and service is now available using VMSES/E. See the PVM 2.1.1 Program Directory for further installation instructions.
- Security Exits are now built in to the PVM module supplied by IBM on the base tape and built through VMSES/E. If the use of these exits are not desired, create a dummy REMOTE AUTH file on PVM's 191 disk.
- German, Kanji, uppercase English, and American English are now built in to the PVM, EFGVIEW, and DVMUSI Modules, and the PVMG Loadlib supplied by IBM on the PVM 2.1.1 base tape and built through VMSES/E.
- CMS APAR VM55319 is required to build the EFGVIEW Module.

© Copyright IBM Corp. 1991,1993

- Provided an MPVM Auto-Signon Facility called MPVMSGN. Refer to *VM/Pass-Through Facility Auto-Signon Reference, SC24-5656* for more details.
- Added PVMG SPOOL command.

Chapter 1. Introduction

This chapter introduces the VM/Pass-Through Facility and its components.

What is PVM?

VM/Pass-Through Facility (PVM) is a communications program used by VM users to access other systems. PVM lets you access, log on, and use another system in defined PVM and System Network Architecture* (SNA*) networks as though your display station is directly connected to that system. All you need is a user ID (*ID* is short for identifier) for the system you want to access. The system you access through PVM is called a **target** system. Once you are logged on to a target system, you may communicate with that system as though you are working on that system.

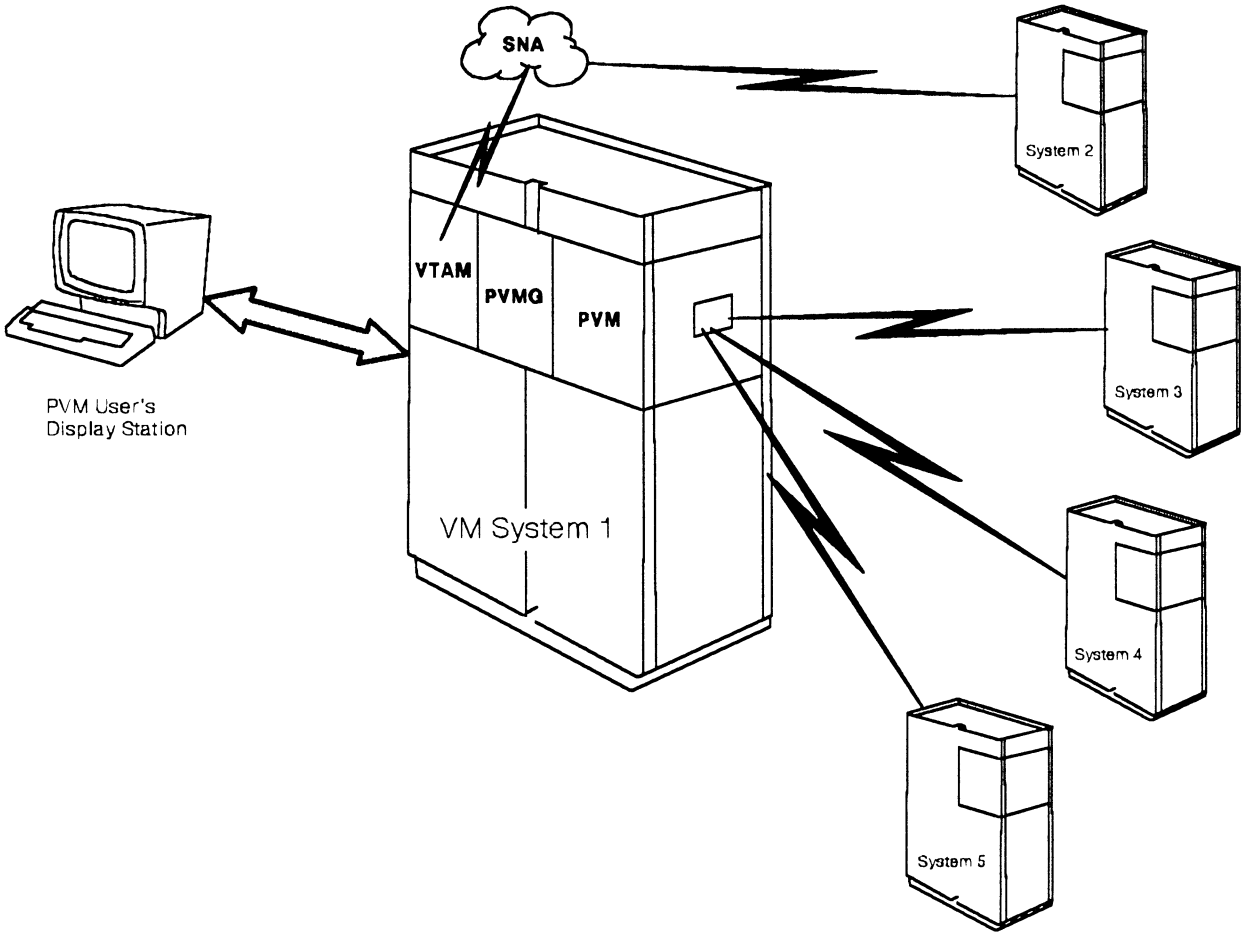


Figure 1. PVM Provides Access to Remote Systems

PVM also provides support for application-to-application communications between VM host applications and personal workstations or other VM host applications. Each VM host application that participates in an application session must have access to a PVM Version 2 virtual machine.

© Copyright IBM Corp. 1991,1993

PVM runs in its own Conversational Monitor System (CMS) virtual machine. As with other virtual machines, PVM is identified by a **user ID**. Because user IDs for PVM can be different from system to system, the term **PVM user ID** refers to a PVM virtual machine.

PVM has four major components:

PVM The PVM component is the server that allows users to pass through to other applications and systems.

PASSTHRU PASSTHRU provides single session support for CMS users.

PVMG The PVMG server runs on its own disconnected virtual machine. PVMG is a Virtual Telecommunications Access Method (VTAM*) application, running on VM's Group Control System (GCS) component, that serves as a communications gateway linking the PVM server machine and the VTAM System Network Architecture (SNA) network. You do not need to install this server if your site does not need to support PVM access to SNA users and applications.

MPVM MPVM provides simultaneous multiple session support for CMS users.

MPVM runs in the CMS end user's virtual machine and supports automation of CMS end-user session activity. It processes MPVM macros written by CMS users and MPVM programmers. It also processes system nickname files that define network connection parameters.

The PVM and PASSTHRU components are installed when you install the PVM product. The MPVM and PVMG components can be optionally installed when you install PVM.

How PVM Enhances Your VM System

PVM enhances your VM system by providing:

- **Interactive communications with remote systems**

  PVM provides interactive access to remote systems and resources that are part of a Distributed Data Processing (DDP) environment. A DDP environment consists of multiple processors that can be physically remote from each other yet are dependent upon one another for various services and resources. These resources may require batch or interactive access from one or multiple processors.

  A VM host or controlling system with PVM installed can access a distributed node for:

  - Remote Operator Console Facility (ROCF) for initializing a remote 4300 processor (IML, if necessary, and IPL)
  - Remote system activities with a distributed system, for example:
    - Program installation and maintenance (VM/Remote Spooling Communications Subsystem Networking (RSCS) can be used for program code transmission.)
    - Display station access to applications, such as the Customer Information Control System* (CICS*), CMS, and Time Sharing Option* (TSO*).

- Remote problem determination through Interactive Problem Control System (IPCS), the Dump Viewing Facility for Virtual Machine/Enterprise System Architecture* (VM/ESA*), the Virtual Machine/Extended Architecture* System Product (VM/XA*) and other aids.

- **Multiple, programmable, simultaneous session capability**

  The MPVM component of PVM allows CMS users to have multiple interactive sessions with different applications at the same time, plus have access to their own host user ID, all from a single terminal. These applications can be on:

  - The same VM system
  - Another VM system
  - A non-VM system that supports 3270 architecture.

  While using an application through MPVM, the CMS users are not aware of any MPVM activities. It appears that they have directly accessed the application.

  CMS users can use MPVM commands to start sessions with different applications. They may also use a toggle key to switch control to any session they have already started.

  Experienced 3270 programmers can write programs using MPVM commands and macros that automate user tasks. For example, an application can be written that simulates a user starting an application, typing data into fields, and then pressing the ENTER key. Many things that a user can do manually at a keyboard can be automated by MPVM.

  Others macros can be written that:

  - Integrate data from several applications onto a user's single display
  - Control how information is transferred from a user's display to an application.

  See the *VM/Pass-Through Facility Programming Reference* for more information about MPVM.

- **PVMG provides a gateway to or from the SNA network**

  PVM users, and users in the SNA network, can access applications through the PVMG server.

- **Application-to-application communications using IUCV protocols**

  PVM supports application-to-application communication through the PVM network between VM host applications, and personal workstation applications. This is referred to as PVM's PCCF support.

- **Low cost networking and concurrent access to multiple systems**

  PVM makes full use of networks from one display station. PVM increases your productivity by providing:

  - Reliable access to many remote systems including other VM systems with PVM installed, or systems that support remote IBM 3270 Control Units
  - Remote access to applications, like CICS
  - Remote access of processors using ROCF
  - Reliable communication across many types of connections, such as connections using a 3088 Multisystem Channel Communications Unit.

Using Advanced Program-to-Program Communication/VM (APPC/VM) and Advanced Communications Function for Virtual Telecommunications Access Method* (ACF/VTAM*), PVM can access other VM systems across links like SNA synchronous data link control (SDLC) lines, X.25 circuits, T1 lines, and the IBM Token-Ring. Dedicated PVM connections are not necessary.

- **Ease of installation**

  After a few changes to your system directory and your system's MAINT user ID, PVM is ready to install. You can use an exec to load the PVM product tape.

  For PVM 2.1.1, VMSES/E is supported for installing and servicing on a VM/ESA system.

- **Ease of operation and use**

  PVM operation is automated by using execs that set up the PVM system to operate the way you configure it.

  PVM is transparent to its users after the PVM session starts. CMS users of PVM can set up their own defaults for the PASSTHRU or MPVM commands, making it easier to start PVM sessions with the options they want.

  General users may want to read the *VM/Pass-Through Facility User's Guide* before using this book. For many tasks, the *VM/Pass-Through Facility User's Guide* contains the information necessary to perform the desired task.

Networking Terms

This section explains some networking terms and their relationship to PVM. Networks consist of systems and Input/Output (I/O) devices connected by communications equipment. In a network, these systems and devices are called **nodes**. The connections between the nodes in a network are called **links**. Figure 2 shows a simple network.

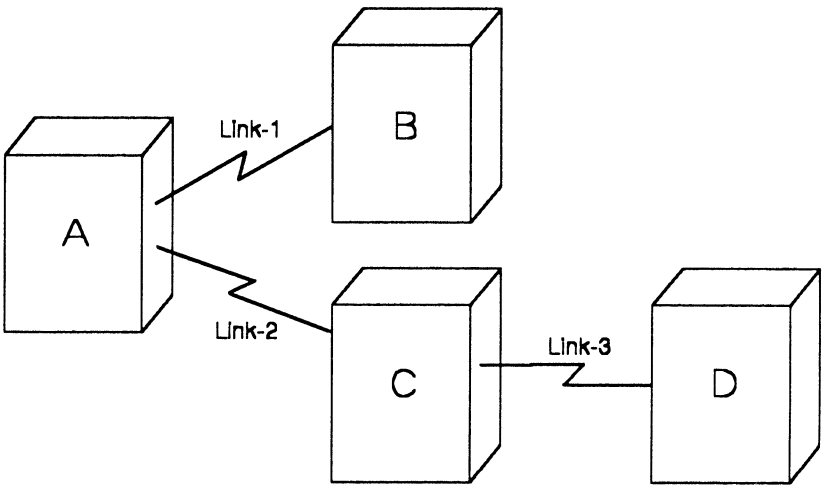


Figure 2. Nodes and Links in a Simple Network

Nodes and links are given names to identify them to users and operators. The terms **node ID** and **link ID** indicate a node's name or a link's name.

The term **local** means the system you are using at that particular time. The term **remote** generally describes a connection to a system made through

communications equipment over PVM-defined links. A **route** is the path from one node in the network to another. An **intermediate** node is a node that is between nodes in a route. For example, in Figure 2 there is a route from node A to node D. Node C is an intermediate node on that route.

Generally, the link ID is the same as the node ID at the other end of the link. Each link includes a programming routine, called a **line driver**, that controls the communications between the nodes on the link. Line drivers use a set of rules, called a **protocol**, to manage communications. There is a different line driver for each protocol that PVM supports. The line driver used for a particular link depends on the hardware and protocol used on the node at the other end of the link.

What You Can Access Through PVM

PVM users can access:

- Systems that support remote IBM 3270 Control Units
- Remote applications, like CICS
- Remote processors using the Remote Operator Console Facility (ROCF)
- Other VM systems with PVM installed
- The VM system on which PVM is installed
- Applications in the SNA network.

Line drivers are used to manage the links to remote systems in a PVM network.Table 1 describes the types of line drivers in PVM.

Table 1 (Page 1 of 2). PVM Line Drivers

Type	Type of Connection	Provides Communications with
BSCA	Point-to-point, nonswitched binary synchronous communication (BSC) line	Another VM system with PVM installed. The target system's PVM uses the CP Logical Device Support Facility to establish communications. This CP facility is described in *VM/ESA: CP Programming Services for 370, SC24-5435.*
CTCA	Channel-to-channel adapter (CTCA)	Another VM system with PVM installed. The target system's PVM uses the CP Logical Device Support Facility to establish communications. This CP facility is described in *VM/ESA: CP Programming Services for 370, SC24-5435.*
327X	Leased BSC lines	Target systems whose control program is not VM with PVM installed. The target system sees the accessing PVM system as a remotely attached 327x Control Unit. For this reason, each non-PVM system to be accessed must include remote 327x support. This support in PVM, called 327x emulation, also provides a means for directing target application output to an IBM 3270 Information Display System printer. 327x refers to 3271, 3274 and 3174 Control Units. These types of control units with a corresponding emulation program provide support to write output to 3270 Information Display System printers or to access target non-PVM systems.
ROCF	Switched, voice-grade telephone line connected by manual dialing	Remote 4300 processors using the Remote Operator Console Facility (ROCF). This line driver gives users the capability to access these systems to perform functions such as remote: • Initialization of a 4300 processor (IML) • Initialization of the VM/SP control program (IPL) • Invocation of a processor's operational display screens for functions like COMPARE/TRACE and ALTER/DISPLAY.

Table 1 (Page 2 of 2). PVM Line Drivers

Type	Type of Connection	Provides Communications with
R3270	BSC 327x Control Unit	Remote 3270 (R3270) terminals and printers connected to a control unit. These devices can be directly controlled by PVM. In an ESA environment, PVM is the only access method for remote 3270 terminals and printers.
3088	3088 Multisystem Channel Communications Unit	Another VM system with PVM installed. This line driver now supports Enterprise System Connection* (ESCON*) channels.
APPC	APPC/VM protocol over a logical path	Another VM system with PVM installed over ACF/VTAM* controlled links.
IUCV	IUCV line driver connection	A PVMG server on the same VM system as PVM.

The different types of systems and connections possible in a PVM network are illustrated in Figure 3.

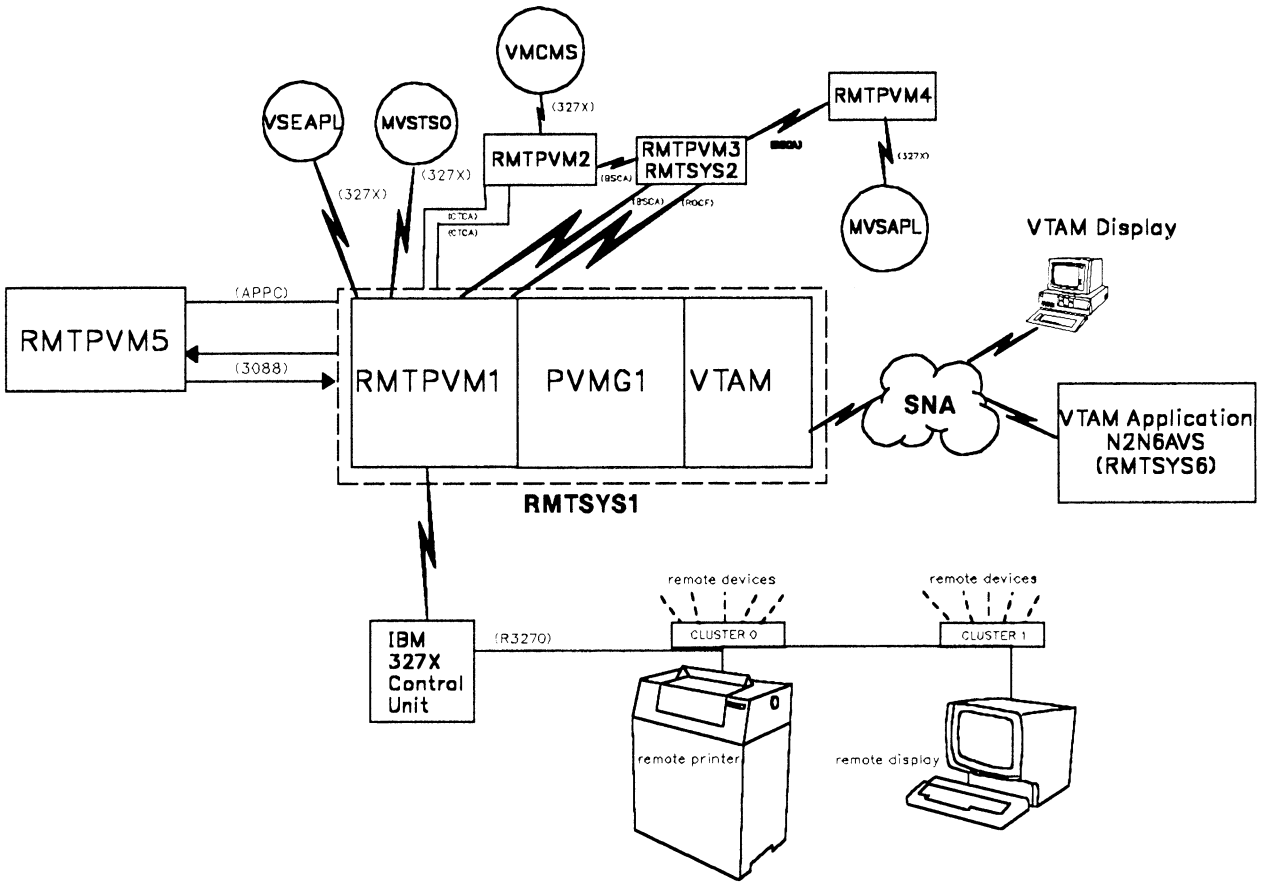


Figure 3. A PVM Network

How You Can Access PVM

There are five ways to access PVM:

1. Access to PVM is through local and remote display stations. Local display stations are attached to a local VM system that has PVM installed. When you turn power on these display stations, you see the VM system's logo screen or the PVM Initial Menu (Initial Menu). The Initial Menu provides direct access to PVM sessions. Access to PVM from the VM system logo is through the CP DIAL command.
2. Remote display stations are attached to an IBM 327x Control Unit. PVM, rather than CP on the system upon which PVM is running, controls the display stations attached to the control unit. When this type of remote display station is powered on, PVM displays a menu called the Initial Menu on the display station's screen. Access to PVM is through this menu.
3. Users can also access PVM by using the PASSTHRU command from a CMS environment on a VM system that has PVM installed on it.
4. CMS users can access PVM using MPVM, which allows them to have multiple simultaneous sessions through PVM.
5. PVM users can access SNA applications through the PVMG server, and users in the SNA network can also access the PVM network through the PVMG server.

Any PVM user can access an application in the SNA network, and any SNA user can access applications in the PVM network through the PVMG server.

An application may access PVM using IUCV protocols to communicate with itself, another virtual machine application on the local system, or another virtual machine application on any PVM node in the network at PVM Version 1.4 or later. Intermediate PVM nodes need not have this support.

Figure 4 on page 8 illustrates the five ways you can access PVM.

Note: Any user in Figure 4 on page 8 can access an application in the SNA network.

Consult the *VM/Pass-Through Facility User's Guide* for instructions on how to access PVM.

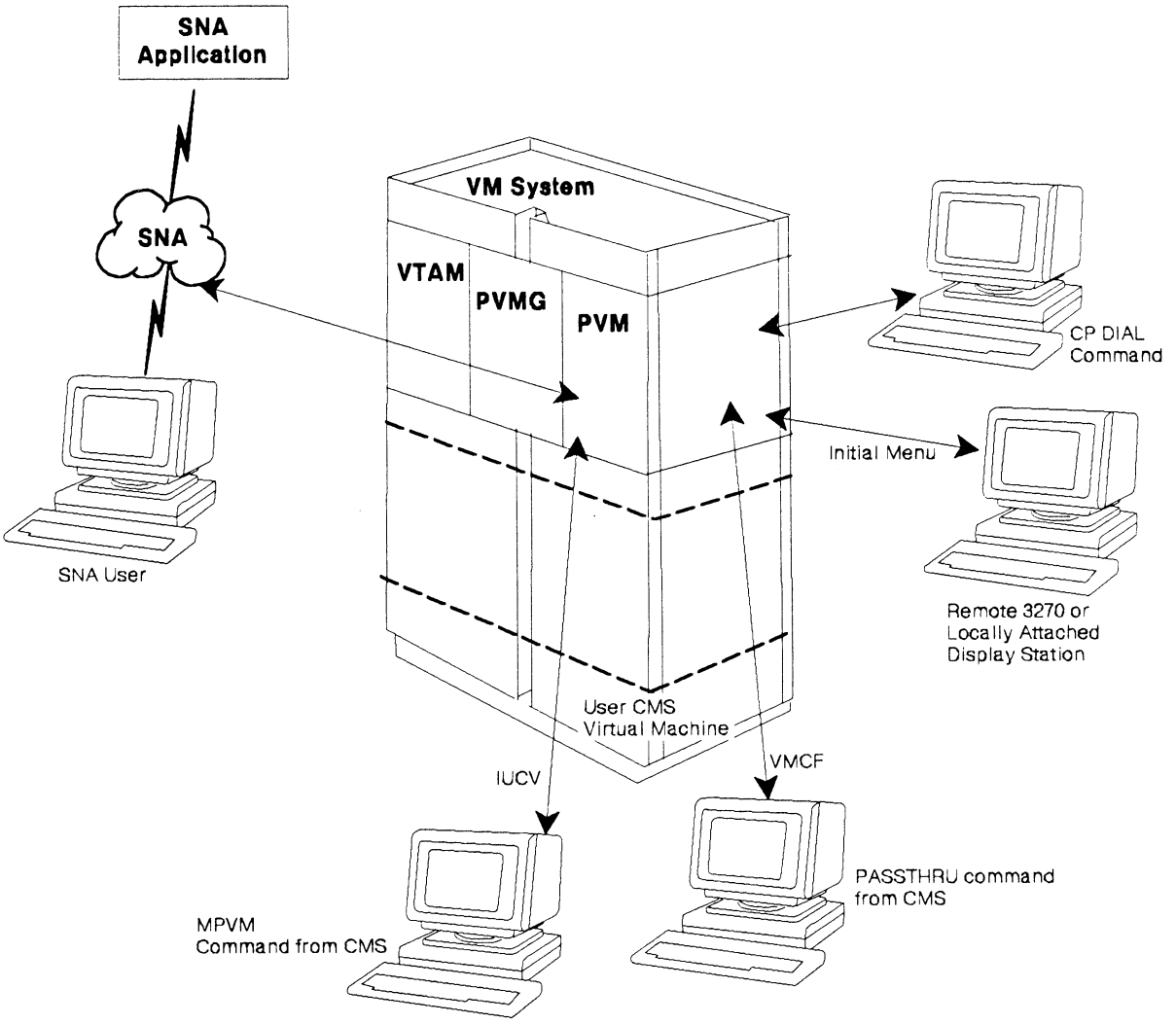


Figure 4. Five Ways to Access PVM to Start a PVM Session

Chapter 2. CVIEW - Product Overview

This chapter is a general introduction to the Cooperative Viewing Facility (CVIEW).

What is CVIEW?

CVIEW is a VM based facility that allows two to twelve users, at separate display stations, to view and share a single VM **interactive session**. An interactive session consists of the activities that occur from the time a user logs on until he logs off. A CVIEW session can be established as either a **consult** or **conference** session. Multiple consult and conference sessions can process at the same time, supporting a total of thirty users at separate display stations.

Users participating in a CVIEW session define their roles and identify the session during "setup mode." Menu and data-entry panels guide them through the setup process. Each menu and data-entry panel has an associated help panel that provides additional information.

A **consult session** supports two users at separate display stations. One user is designated as the **consultant** and the other user is the **recipient**. The consultant is an expert in a particular subject area and provides help or assistance to the recipient. In a consult session, the keyboards of both users are "active," allowing either user to enter data or commands.

A **conference session** supports from two to twelve users at separate display stations. During setup mode, one user is designated as the **moderator** and all other users participating in the conference are **attendees**. The moderator has an active keyboard and control the conference session. Attendees have "inactive" keyboards so they cannot enter data or commands. The moderator can activate an attendee's keyboard by assigning **presenter** status to him. The moderator can drop participants from the conference, and attendees can remove themselves from the conference.

In a CVIEW session, users view identical screen images at the same time. The output from host and application processing is always displayed. CVIEW also allows the display of user input before the input is processed by the host or application. This **display-input** option is activated by the consultant and the moderator. After the display-input option is activated, any user with an active keyboard can "direct" his input to all display stations in the session.

Figure 5 on page 10 shows how display stations in consult and conference sessions relate to each other and to the CVIEW virtual machine during an interactive session.

© Copyright IBM Corp. 1991,1993

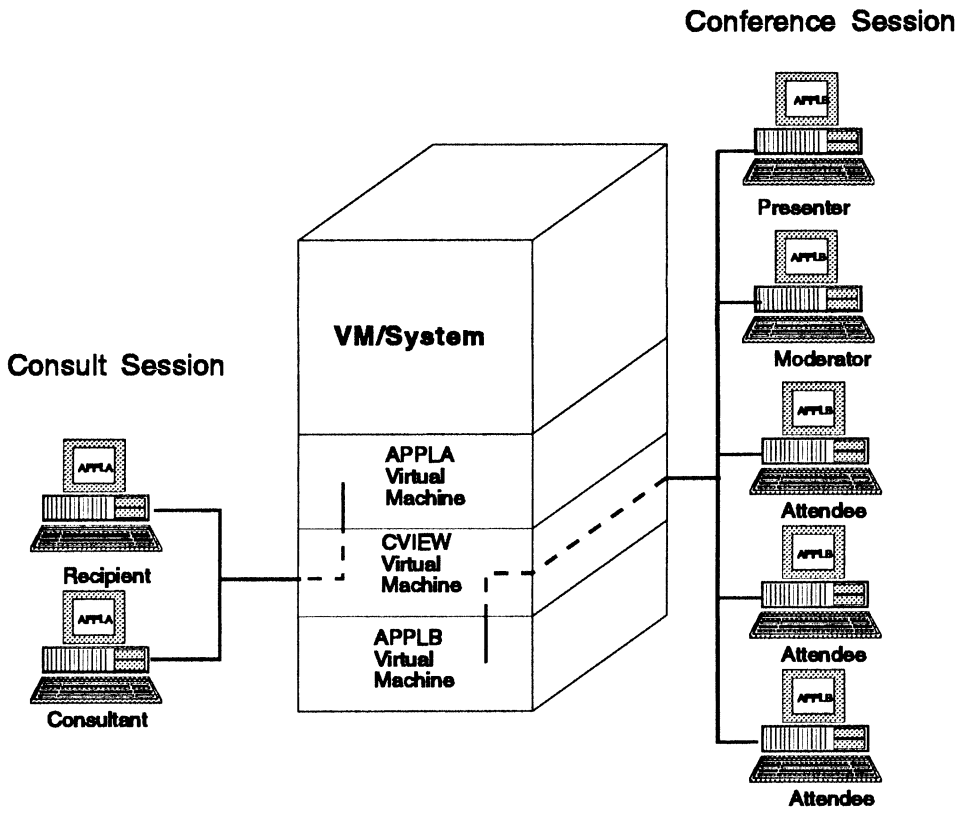


Figure 5. CVIEW Sessions and the CVIEW Virtual Machine

Modes of Operation

This section describes the different modes of operation for CVIEW.

Setup Mode

Setup mode starts after a user enters 'DIAL CVIEW' from his display station. The first panel, the **HOME** panel, is displayed to the user, indicating the beginning of setup mode. The user selects his role (recipient, consultant, moderator, or attendee) for the session from the HOME panel. See Appendix K, "CVIEW Product Panels" on page 457 for descriptions of all the CVIEW panels.

As setup mode continues, additional panels are displayed to the user to further define the session in which he will participate. For a consult session, setup is complete when the recipient and consultant have responded to all the setup panels required to define the session. For a conference session, setup is complete when the moderator starts the conference session. When setup mode is complete, the CVIEW session starts.

Session Mode

Before **session mode** starts, the CVIEW virtual machine associates all session participants with a single interactive session. This allows the CVIEW virtual machine to provide identical screen images to the participants at the same time.

At the start of a session, the VM logo of the system where the CVIEW virtual machine runs is displayed to each session participant. In session mode only one user, with an active keyboard, logs on to the system. This user can log on any virtual machine user-ID defined in the system's directory or may Pass-Through to another system node if desired. Session mode continues until a user with an active keyboard logs off the system. After the LOGOFF command has been processed, all session participants return to setup mode, from which they can leave CVIEW or define another CVIEW session.

Figure 6 on page 12 shows different types of interactive sessions in which users can participate using CVIEW.

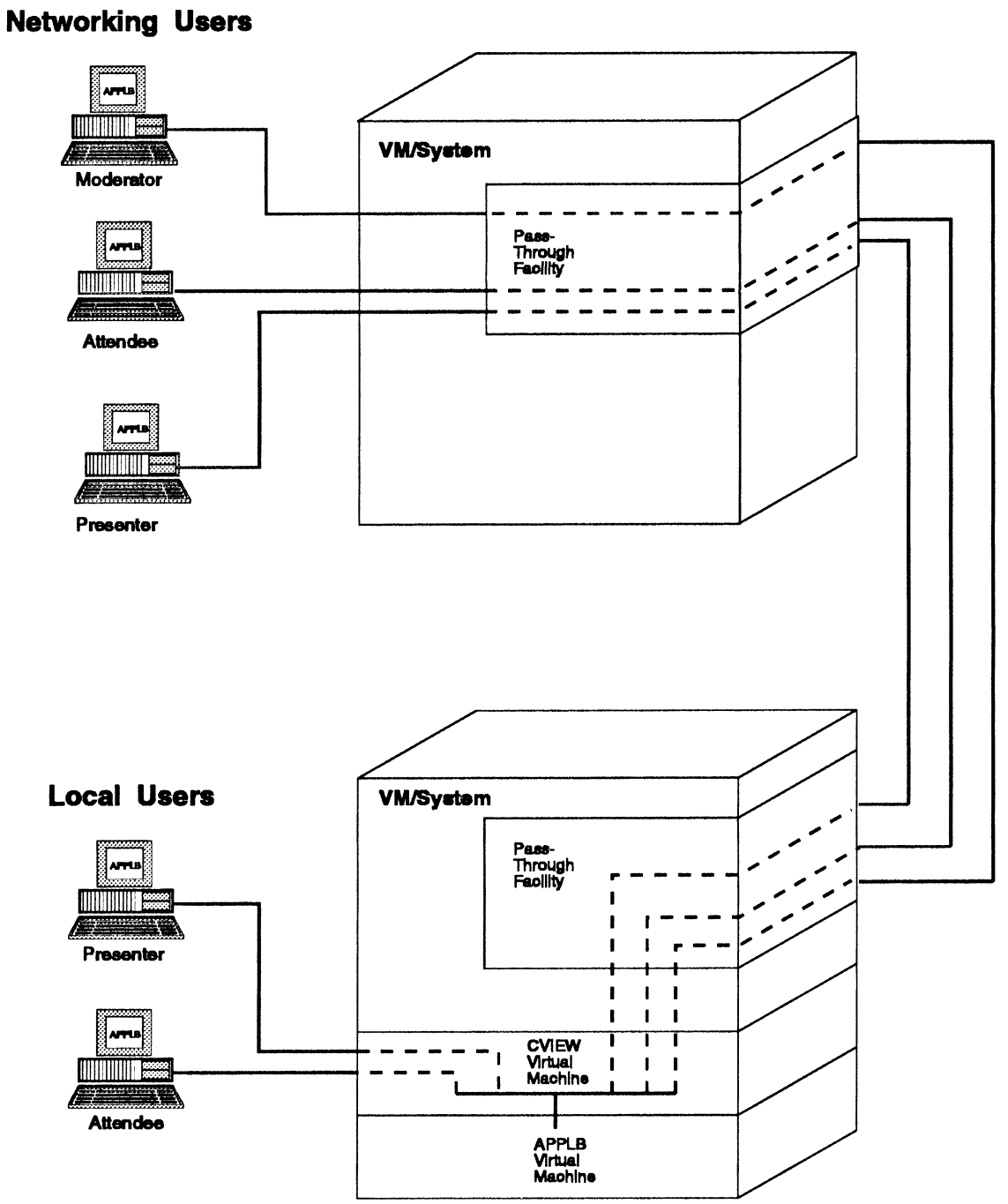


Figure 6. Different Types of Interactive Sessions Using CVIEW

Notify Mode

Notify mode occurs when a consult or conference session is automatically suspended by the CVIEW virtual machine in order to display the **NOTIFY** panel. The NOTIFY panel advises a user that another user is sharing his interactive session and associated resources. The NOTIFY panel is displayed only if its use was selected during customization. Notify mode can occur at two different time intervals during a session. The first panel is displayed a specified number of

seconds after session mode begins. Subsequent panels are displayed at preset intervals as the session progresses.

When a NOTIFY panel appears on a user's screen, he must press the ENTER key to provide a positive response before the session can continue. Pressing the HOME or QUIT PF key in response to the NOTIFY panel terminates the entire consult or conference session. NOTIFY panels can be displayed to all session participants or only to participants who have entered data or commands since the last NOTIFY panel was displayed. All participants receiving a NOTIFY panel must respond to it.

Control Mode

The consultant and moderator can suspend a session to enter **control mode** by pressing the Program Attention 1 (PA1) key. The consultant then sees a **CONSULT SESSION CONTROL** panel. The moderator sees a **CONFERENCE CONTROL** panel.

When the **consultant** activates control mode, he can assign or change a PF key for the display-input option or request CVIEW to forward the PA1 key function to the host or an application program for processing. After the consultant selects a PF key for the display-input option, he presses the ENTER key and the consult session resumes. The PF key assigned for the display-input option can be used by both the consultant and the recipient.

When the **moderator** activates control mode, he can also assign or change a PF key for the display-input option or request CVIEW to forward the PA1 key function to the host or an application program for processing. The PF key assigned for the display-input option can be used by the moderator and any presenters.

In addition, the moderator can request the **CONFERENCE PARTICIPANTS** panel, which when displayed allows the conference session to resume without the moderator if a presenter also exists.

The moderator can assign presenter status to any number of attendees listed on the CONFERENCE PARTICIPANTS panel. Likewise, attendee status can be assigned to any presenters listed. The moderator can drop participants from the conference as necessary. After the moderator changes the status of participants, he must press the ENTER key to process the new assignments. The moderator returns to the conference in progress by pressing the SESSION PF key from the CONFERENCE PARTICIPANTS panel.

Technical Overview

This section contains more detailed information about the way CVIEW operates. It is intended for the system programmer or administrator who must install, customize, and maintain the program.

Accessing the CVIEW Virtual Machine

CVIEW processes as a CMS application in a disconnected virtual machine. The CVIEW virtual machine is logged on by the person designated as the CVIEW operator. The operator loads the DXGCVIEW exec to start CVIEW. After the CVIEW virtual machine is operational, the operator can enter a CP DISC command.

Users who want to establish a CVIEW session do not log on the CVIEW user-ID. Instead, they enter the 'DIAL CVIEW' command. After a user dials the CVIEW virtual machine, he is in setup mode. The CVIEW virtual machine communicates with his display station as a **GRAF device**, rather than as a virtual machine console.

During setup mode, a **GRAF address** (501 through 530) is assigned to each user's display station. When session mode starts, CP creates a **logical device** for the CVIEW session. The CVIEW virtual machine associates the GRAF addresses in a session with the address of the logical device. During session mode, the **logical device address** is used by the CVIEW virtual machine to communicate with each user's display station as though a single virtual machine console existed for the interactive session.

Figure 7 shows how the operator and end users access the CVIEW virtual machine to perform their specific tasks.

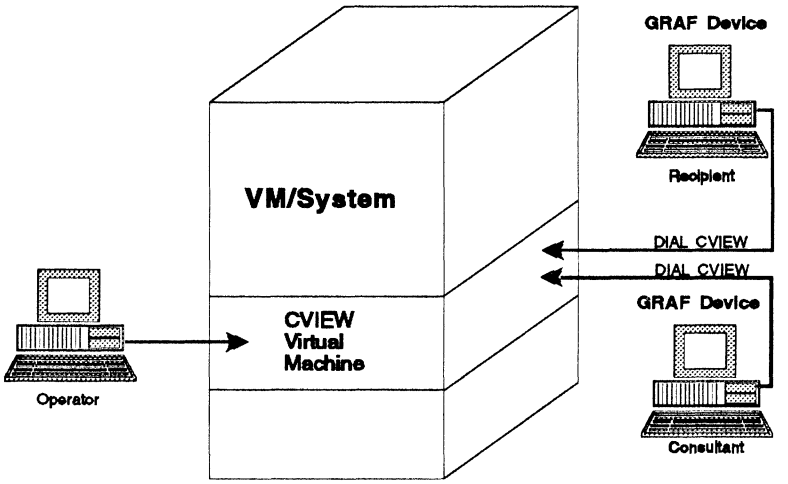


Figure 7. Accessing the CVIEW Virtual Machine

Networking Sessions

A CVIEW **networking session** includes one or more users who have accessed the CVIEW virtual machine through the IBM VM/Pass-Through Facility. These users are called **networking users**.

The IBM VM/Pass-Through Facility allows display stations from another hardware system to access the target system node where the CVIEW virtual machine is running. A Pass-Through session can be established by issuing the CP DIAL command from the CP environment, by issuing the PASSTHRU or MPVM commands from a virtual machine console in the CMS environment, or by using the PVMG bridge.

Figure 8 on page 15 shows how a networking session relates to the CVIEW virtual machine.

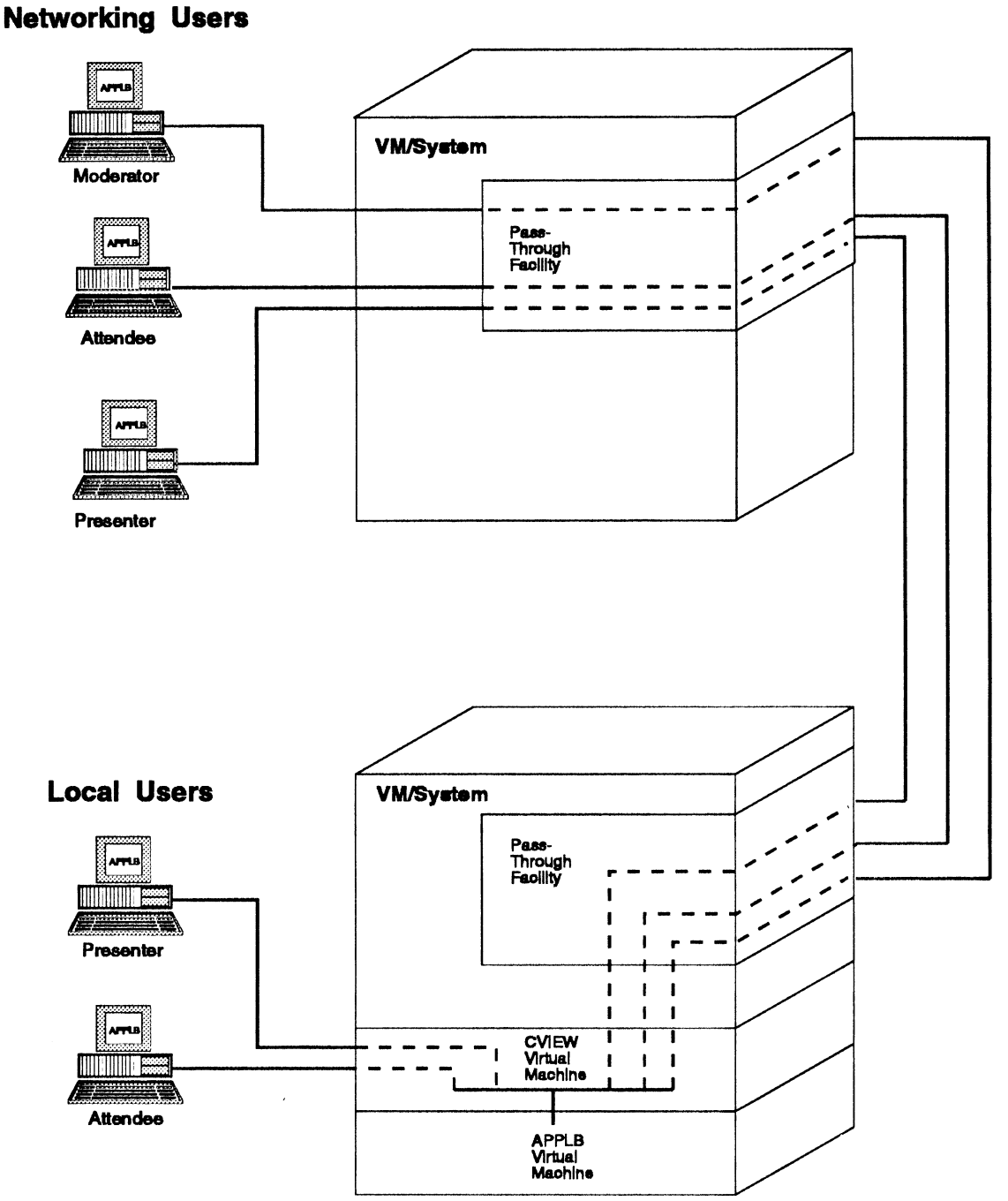


Figure 8. An Example of a Networking Session

Networking Considerations

After a user initiates a Pass-Through session, a logical device is created on the target system node for his Pass-Through session. This establishes communications with the target system node as though the user's display station were locally attached. (The CVIEW virtual machine also uses a logical device for communications in a consult or conference session.) The CVIEW virtual machine associates all display stations (GRAFs) in a session with a single logical device.

While CVIEW can support as many as 30 networking users, each one must wait in setup mode until all other networking users have dialed the CVIEW virtual machine. To accommodate additional consult or conference sessions that include networking users, you can create multiple CVIEW virtual machines on the same system. CP allows multiple virtual machines (for example, Pass-Through and CVIEW) to simultaneously create logical devices. Refer to the *VM/ESA: CP Programming Services, SC24-5520* for the Logical Device Support Facility limitations.

PASSTHRU Considerations

The PASSTHRU command, which can be issued by Pass-Through users from within a virtual machine, supports a feature called temporary disconnect. This feature allows a Pass-Through user connected to a target system use the PASSTHRU command to temporarily disconnect from the Pass-Through session with that system and return control to local CMS. When the desired interaction with the local CMS system is complete, the user can resume the session with the target system by reissuing the PASSTHRU command.

If the session with the target system is not resumed within an installation-specified time limit, Pass-Through terminates the session.

Networking users who have issued the PASSTHRU command might cause the CVIEW virtual machine to temporarily suspend if the Pass-Through temporary-disconnect feature is used. The CVIEW virtual machine would be suspended until the session loaded using the PASSTHRU command is resumed or the installation-specified Pass-Through time limit expires (the default is 20 minutes).

Other conditions can result in similar problems. Any condition that causes the Pass-Through session to end may result in a temporary disconnect. A preventive step to minimize the time that the CVIEW virtual machine is suspended under these conditions would be to re-define the Pass-Through limit for temporary disconnects to the minimum.

Display-input Option

Any consult or conference session participant whose keyboard is active can use the display-input option. In setup and control modes, a consultant or moderator can assign a PF key as the display-input key. After input data or commands are keyed in, the user presses that PF key and the input appears on all display stations in the session. The input is not processed by the host or application program until the ENTER key is pressed.

The display-input PF key can be pressed repeatedly to show a series of steps or corrections. When the cursor is moved with a cursor movement key, pressing the display-input PF key shows the new location of the cursor on all screens.

The display-input PF key shows only screen input changes that are processed by the Read Modified 3270 data-stream command. The Read Modified command does not send null codes back to CVIEW. Therefore, the results of pressing the ERASE EOF key do not show up when a user presses the display-input PF key.

The use of the display-input PF key is optional; the consultant or moderator can specify NONE instead of a PF key number. If a PF key is assigned, that key

functions only as the display-input key in session mode. All other PF keys operate the way they were defined to your system and application program.

Accommodating Different Display Station Types

CVIEW supports a variety of display station types and models. Because CVIEW displays identical screen images to all participants in a session, it must assign one display station type and model to each session when the session starts. The display station types and models CVIEW can assign to a session are 3277-2, 3278-2, 3278-3, 3278-4, or 3278-5. See "Device Type and Model Assignment" on page 136 for the specific display stations supported by CVIEW.

After CVIEW assigns a display station type and model to the session, the CVIEW virtual machine reads and writes information to all display stations in the session as if they were that same type and model. A user can only join a conference session after it has been started by the moderator if his display station type and model is compatible with the type and model already assigned to the conference session. See "Joining a Conference Session in Progress" on page 136 for details.

Messages

CVIEW issues information, eventual action, and action messages to the user during setup mode as required. The messages explain responses if required of the user.

The information displayed to the CVIEW virtual machine console depends on the setting of EMSG for the virtual machine. Messages displayed to the CVIEW virtual machine console appear with either the message number and text, text only, or do not appear at all. It is recommended that EMSG be set to TEXT for the CVIEW virtual machine.

For detailed information on CVIEW messages, see *VM/Pass-Through Facility Messages,* SC24-5648.

Part 1. Administering, Installing, and Operating PVM

Chapter 3. Administrative Considerations 21
What's New In PVM Version 2 21
Installation Requirements 22
Device Support 23
Storage Considerations 25
How Much Storage is Need for Your PVMG Server? 29
Performance Considerations 29
Security Considerations 29
Restrictions 32
Migrating from PVM 1.4 36
Using PVM Version 2 in a VM/ESA or VM/XA Environment 37

Chapter 4. Installing PVM 2.1 41
Step 1. Add Directory Entries for the PVM and PVMG Operators 41
Step 2. The MAINT (or Installer's) User ID 44
Step 3. Getting Disks for PVM, PVMG, and MPVM Files 44
Step 4. Load the Tape Files to Disks 46
Step 5. Set Up Autolog Statements for PVM and PVMG 51
Step 6. Set Up the PVMG Server Virtual Machine 52
Step 7. Add PVMG to GCS 52
Step 8. Add VTAM Definitions 53
Step 9. Apply Service 54
Step 10. Verify Initial Installation 54

Chapter 5. CVIEW Installation Planning 57
CVIEW Virtual Machine Requirements 57
CVIEW Directory Entry 58
CVIEW Operating Environment 58
Multiple CVIEW Virtual Machines 59

Chapter 6. Defining Your PVM Configuration 63
Planning a Network 63
Creating the Configuration File 66
Creating a CMS PROFILE EXEC for PVM 79
Creating a CMS Communications Directory 80
Creating a PVM Profile Exec 81

Chapter 7. Defining Your PVMG Configuration 85
Planning Your PVMG Configuration 85
Planning the PVMG Server's Role in the PVM Network 86
Tailoring the PROFILE GCS File 93

Chapter 8. Developing NAMES Files for MPVM 95
Creating Entries for Applications 95
Defining Configuration Settings—:nick.?CONFIG 98
Creating Other Special Entries 98
Sample MPVMSYS NAMES File 99
Verifying the MPVMSYS NAMES File 100
Considerations for Individual User NAMES Files 100
Summary of MPVMSYS NAMES Contents 101

© Copyright IBM Corp. 1991,1993

Chapter 9. Operating the Network 103
Starting PVM 103
Querying the System 107
Defining a Link Dynamically 109
Displaying the Path of a Session 110
Sending Messages 111
Issuing CMS Subset Mode Commands 111
Quiescing, Resuming, and Dropping Links, Devices, and Users 112
Using the PVM Programmable Operator (PVMPROP)/Command Response Interface (CRI) 114
Stopping PVM 114
Starting and Stopping Printer Sessions 115
Starting PVMG 117
Querying PVMG 117
Displaying the Path of a PVMG Session 118
Sending Messages 118
Dropping Users and Nodes 119
Stopping PVMG 120

Chapter 10. Using CVIEW 123
Starting the CVIEW Virtual Machine 123
CVIEW Commands 124
CMS 124
CP 125
DROP 125
HELP 126
PASSWORD 126
QUERY 127
SHUTDOWN 129
STOP 130
Establishing a Session 130
Managing a CVIEW Session 133
CVIEW PROFILE EXEC 137
CVIEW Log File 137

Chapter 3. Administrative Considerations

This chapter contains information you need to consider before installing PVM Version 2. For users of PVM Version 1.4, this chapter lists new features and other considerations that could impact migration to PVM Version 2.

What's New In PVM Version 2

PVM Version 2 provides new functions, components, and enhancements. These include:

- The MPVM component
  - Provides simultaneous multiple session support
  - MPVM users can enter commands to their host user ID while using another PVM session by using MPVM's escaped command support (escaped commands are those commands that are sent to the MPVM user's host user ID for execution)
  - MPVM end-user programmability to automate everyday tasks.
- The PVMG component provides:
  - A gateway that allows PVM users to connect to SNA applications through VTAM
  - Users in the SNA network to access the PVM network
  - Simulation of Primary Logical Unit (LU) type 2, and non-SNA 3270 Secondary LU type 2
  - An IUCV line driver for communication with the PVM component.
- The PVM component includes:
  - A new IUCV line driver for communication with the PVMG component
  - Support for the IBM ESCON channel connection through the IBM 3088 1F Control Unit to another PVM server using PVM's 3088 line driver
  - Sample access security exits
  - Support for graphic and printer devices attached to an ESCON 3174 control unit (models 12L and 22L)
  - New and changed PVM commands including:
    - SET
    - QUERY
    - DROP
    - DEFINE

    For a description of these commands, see Appendix C, "PVM Commands."
  - New and changed PVM configuration file records including:
    - MAXIDLE
    - MAXSESS
    - MAXSUSP
    - MAXDISC
    - MAXUSERS

© Copyright IBM Corp. 1991,1993

- LINK

For a description of these records, see Appendix D, "PVM Configuration File Records."

There are also new commands and configuration file records to support the PVMG component of PVM. See Appendix E, "PVMG Commands" on page 375 and Appendix F, "PVMG Configuration File Records" on page 401 for information regarding these new commands and configuration file records.

Installation Requirements

- Each node in a PVM network must have a unique node ID, including each PVMG server. If more than one link to a non-PVM system node is required, that node must be assigned a unique node ID for each link. You can have a group of links between two PVM nodes (not including the PVMG server), but you cannot have more than one group between the two PVM nodes.
- PVM Version 2 is supported on any of the following systems, at the stated release or service level or higher:
  - VM/Enterprise System Architecture (VM/ESA) Release 1.1 or higher
  - VM/ESA Release 1.0 (ESA and 370 Feature)
  - VM/SP Release 5
  - VM/SP HPO Release 5
  - VM/SP Release 6
  - VM/HPO Release 6
  - VM/XA SP Release 2.1

  Note: PVM 2.1.1 is only supported on VM/ESA releases.

  To use the TRACE command with the SYSTEM ETRACE options, PVM Version 2 must be running on VM/SP Release 6 or VM/ESA Release 1.0 (370 Feature).
- APPC/VM considerations:
  - For PVM to define APPC links, Advanced Communications Function for VTAM (ACF/VTAM) Version 3 Release 2 (or higher) and the APPC/VM VTAM Support (AVS) component of VM/SP Release 6 or VM/ESA Release 1 (or higher) must be installed.
  - The AVS component of VM provides the gateway support needed to communicate over a SNA-defined network. APPC/VM uses AVS to access this VTAM controlled link. See *VM/ESA Connectivity Planning, Administration, and Operation, SC24-5448* for detailed information on installing AVS.
- For 3088 type devices, specify SHARED=NO on the CNTLUNIT macro in the IOCP Gen.

  Class B privilege must be assigned to the PVM server when:
  - The PVM configuration file contains a MSGNOH record
  - Warning messages are allowed to be issued to PVM users
  - The PVM configuration file contains CLPORT records for remote printers
  - Lines or 3270 devices are attached by ATTACH or NET ATTACH commands either in the PROFILE PVM file or from the PVM user ID.

  Class B privilege must be assigned to the PVMG server when the PVMG configuration file contains a MSGNOH record.

327x emulator lines must have one of the following:

- The Station Select feature installed if a 270X Communications Controller (CC) is used
- The tributary addresses generated if the 370X CC is used
- The tributary addresses specified if the Communications Adapter (CA) on the 4321/4331 is used.

Almost all data in the PVM and PVMG configuration files must be entered in uppercase. The session end string specified on the SESSTERM record is the only piece of data that can be entered in mixed case. PVM configuration file records must be entered in the sequence illustrated in Figure 17 on page 70. PVMG configuration file records do not need to be entered in a particular order. Port numbers and types defined in the PVM configuration file must be consistent with the definition of these ports at the remote node. Make sure that all 327x emulator lines have the control unit address (0-31) defined through:

- Inclusion of the Station Selection feature if a 270X Communications Controller is used
- Generation of tributary addresses (TADDR= parameter on the LINE macro) if the 37XX Communications Controller (37XX refers to the IBM 3704, 3705, and 3725 Communications Controllers) is used
- Specification of tributary addresses (TADDR= parameter) on the corresponding display panel if the CA on the 4321/4331 is used. (For 4331, the emulator line is a multidrop from the host system.)

The following values should be used to specify the control unit address (0-31):

For Control Unit	Use Value	For Control Unit	Use value
0	40	16	50D0
1	C141	17	D1
2	C2	18	D2
3	C3	19	D3
4	C4	20	D4
5	C5	21	D5
6	C6	22	D6
7	C7	23	D7
8	C8	24	D8
9	C9	25	D9
10	4A	26	5A
11	4B	27	5B
12	4C	28	5C
13	4D	29	5D
14	4E	30	5E
15	4F	31	5F

Device Support

Access to PVM may be gained through an IBM 3276 Control Unit Display Station (Model 2, 3, or 4), or through any of the following IBM Display Stations:

3277-2
3278-2, 3, 4, or 5
3279-2 or 3

3290-220, 230, T30.

The display stations listed in Table 2 are supported by PVM when they are specified as the models given in the right column. Extended data streams are supported for sessions with PVM hosts.

Note: The extended data streams are not supported by PVM when the session is with a non-PVM host (through the 3270 emulator line driver).

Table 2. Display Stations Supported by PVM

Display	**Supported as a**
3178 C1, C2, C3, C4 3191	3278-2
3180-1 3290 220, 230, T30	3278-2, 3, 4, or 5 **Note:** MPVM supports any session screen size (defined by ROWs and COLumns variables) up to the 3290 in ALTERNATE (62 rows x 160 columns) mode.
3179 G1, G2	3279-3
3193-1, 2	3278-2, 3, or 4
3194 H20, H50	3279-2
3179-1 3270-PC 3270-PC/G, GX, 3270-AT/G, AT/GX PC, PC/XT*, PC AT*, PC AT/370 PS/2* 5540, 5550, 5560 RT Personal Computers RISC System/6000* (RS/6000*) workstations	3278-2 or 3279-2
Note: The PC, PC/XT, and PC AT are supported if they have the 3278/9 emulator card installed.	

The 3278 and 3279 Display Station models are also supported in 3277 compatibility mode. The minimum supported screen size is 24 lines of 80 characters each. These display stations can be attached to any IBM 327x Control Unit supported by VM as a display station control unit. PVM emulates an IBM 3271 (Model 2) or 3274 (Model 1C) Control Unit when accessing a system (other than another VM system with PVM installed or a 4300 processor) over any BSC line supported by that system. These accessed systems must support the remote IBM 3271/3274 Control Unit.

PVM also provides printer support to non-PVM and PVM systems for the following 3270 Information Display System printers:

- 3284-1 and 2
- 3286-1 and 2
- 3287-1, 1C, 2, and 2C
- 3288-2
- 3289-1 and 2
- 3262-3 and 13 (when defined as 3289).

The following printer devices are supported by PVM when specified as 328X, or a model equivalent to one of those listed above.

Note: Extended data streams are not supported by PVM when the printer session is with a non-PVM host (through the 3270 emulator line driver). PVM also does not support the Intelligent Printer Data Streams:

- 3268-2, 3268-2C
- 4214-1
- 4224-201, 202, D12, D20, 2E2, 2C2
- 4234-1 (compatible with a 3262)
- 5210-G1, G2
- 5227-11
- 5553.

PVM display and printer support includes the special features of extended color, extended highlighting, and programmed symbols for PVM to PVM sessions. For PVM to non-PVM sessions, these special features are not supported.

Storage Considerations

When planning your PVM installation, there are several considerations concerning the storage needed for PVM operation. This section covers some of these considerations.

DASD Requirements

Note: For PVM 2.1.1 DASD requirements, see the Program Directory supplied with the product tape.

Table 3 shows the approximate auxiliary direct access storage space:

Table 3. DASD requirements (in cylinders)

Minidisk	3330	3340	3350	3380	3390	3375	3310/3370 (in 1 K blocks)
PVM System (36E)	16	38	8	6	5	10	5580
PVMG System (38E)	6	13	3	2	2	3	1860
MPVM Common Disk (38F)	13	32	6	5	4	8	4650
Base TEXT Staging (49E)	37	88	18	14	12	22	17000
Source Staging (39E)	85	200	40	32	27	49	29000
Service Staging (29E)	19	44	9	7	6	11	6510
System Extension (19E)	6	13	3	2	2	3	1860
IPCS Text (193)	3	7	2	1	1	2	930
Mixed-case American English HELP (19D)	13	32	6	5	4	8	4650
NLS feature tape HELP (each language, 401, 402, 405)	13	32	6	5	4	8	4650

Allow for equivalent space on other supported DASD devices.

Be sure that adequate auxiliary storage space is provided for daily PVM operations (specifically the PVM system disk). For example, additional storage is required for the AUDIT records if that option is used.

You can reduce the virtual storage requirements by removing parts of the PVM program. See "Tailoring the PVM Module" on page 201 for instructions on how to do this.

Storage Needs in the Default Configuration

PVM (the PVM MODULE) is designed to run in a minimum of 5M virtual machine when run with certain supplied defaults. These defaults allow an installation to run with:

- 100 users
- 500 sessions
- 4 links
- 50 blocks.

PVM has three configuration file records that help you reserve storage for your PVM system. These are the BUFFERS, EXT, and I/O records.

BUFFERS Record

The BUFFERS record specifies how many 4096-byte pages of virtual storage will be reserved for use as internal PVM buffers. The number of buffers required by the system depends on the number and type of active line drivers, the number of active PVM users, and the number of commands being processed by PVM. If you do not put a BUFFERS record in your PVM CONFIG file, PVM uses a default of 30 pages. See "BUFFERS" on page 350 for more information on the BUFFERS record.

EXT Record

The EXT record specifies how many entries will be created to handle various external interrupts. One entry is used by the system as the default VMCF message handler, and one entry is used by the system to handle SMSG interrupts. (VMCF stands for Virtual Machine Communication Facility, a CP function that provides a method of communication and data transfer between virtual machines operating under the same VM system.) In addition, there must be one entry for each active user accessing PVM from CMS (by using the PASSTHRU command). There must also be one entry for each user gaining entry to the local CP system as a target node. Also, the APPC type line driver uses 2 entries for each active line driver. The IUCV type line driver uses one entry for each active line driver to PVMG. MPVM uses 2 entries for each active user. Each entry requires 16 bytes of storage. If you do not put an EXT record in your PVM CONFIG file, PVM uses a default of 50 entries.

I/O Record

The I/O record specifies how many entries will be created to handle I/O interrupts. One entry is used by the system to handle attention interrupts. In addition, there must be one entry for each active line driver (START LINE), one entry for each active attached display support task (START GRAF), and one entry for each active attached printer support task (START PRT). Each entry requires 16 bytes of storage. If you do not put an I/O record in your PVM CONFIG file, PVM uses a default of 50 entries. See "EXT" on page 354 for more information about the EXT record.

Modifying Your Storage Needs

You can change the amount of PVM storage by using the BUFFERS, EXT, and I/O configuration file records. Appendix D, "PVM Configuration File Records" contains algorithms you can use to compute the values for these records based on your PVM system configuration.

This section describes things you should consider before you install PVM, including how much storage:

- The MPVM component requires in a user's virtual machine
- Is required by the PVM and PVMG server virtual machines.

How Much Storage Is Needed in Each User's Virtual Machine for MPVM?

The amount of storage that MPVM requires in each user's virtual machine depends on whether shared segments are used. Whether shared segments are used or not, remember that the storage figures shown here are only estimates.

Virtual Storage Required When Using Shared Segments

By building a shared segment, you have just one copy in storage, rather than individual copies in each user's virtual machine. For example, you can create a shared segment that contains all of MPVM's CMS end user interface code.

If you generate an MPVM shared segment, it can contain the following items:

- Code for the MPVM user module that contains the MPVM user interface
- The message repository, which contains text for all messages. (The default language for these message texts is American English)
- The command table, which contains MPVM variable names and keywords for use on MPVM user commands.

See "Installing MPVM in a Shared Segment" on page 203 for information about installing MPVM in a shared segment.

In addition, you can add commonly-used MPVM macros and commands to the CMSINST installation shared segment. Refer to "Adding Common MPVM Commands and Macros to Your Installation Shared Segment" on page 205.

Assuming you install MPVM in a shared segment, here is how to approximate the storage that an MPVM user requires for each virtual machine:

1. Estimate the size of dynamic working storage obtained during execution. This is difficult to predict. Working storage is obtained for a variety of user-initiated activities. However, storage is definitely required for the following:
   a. The screen size, multiplied by 2 for screen-image buffers. (Every MPVM terminal screen has two display *planes*; this is described in the *VM/Pass-Through Facility Programming Reference* book.) For example, for a 24x80 screen size, your total is 3840 bytes (2*24*80=3840, round to 4K).
   b. If any target application accessed through PVM loads programmed symbol sets, add storage for a third screen-image buffer. For example, for a 24x80 screen size, add 1920 bytes (24*80=1920, round to 2K).
   c. A maximum-supported screen size (3*62*160=13020 bytes, round to 30K) for a console read buffer, which may be needed by target applications.

d. 16K for a basic dynamic storage area

e. Anywhere from 32K to 160K for execution storage buffers for the portion of MPVM that still resides in the user's virtual machine. Use 85K as an average.

Here is a typical example for the user virtual machine storage required when (1) using an MPVM shared segment and (2) storing most MPVM macros and commands in your installation shared segment:

```
  A 32x80 terminal (32*80*2 = 5120, round up)  =    6K
+ No programmed symbol sets loaded             =    0K
+ Storage for a console read buffer            =   30K
+ Dynamic storage area                         =   16K
+ Other user virtual machine requirements      =  137K
                                                 ------
                                                   137K
```

Storage Required When Not Using a Shared Segment

If you are not using any shared segments, you need the storage required in the previous section, *plus* the following:

- Approximately 198K for a private copy of the MPVM user module (including the message repository)
- About 100K for the commonly used MPVM commands and macros which are automatically loaded into storage.
- About 4K for the command language (variables and keywords) table

Here is a typical example for the user virtual machine

```
+ Storage required from previous section       =  137K
+ Private MPVM user module                     =  198K
+ PVM commands and macros                      =  100K
+ Message repository and command table         =   22K
                                                 ------
                                                   457K
```

So by using shared segments for MPVM, you can save about 320K per virtual machine.

How Much Storage Is Needed for Your PVM Server?

The PVM server virtual machine typically runs as a disconnected server virtual machine. It initially directs all PVM connections and maintains screen image buffers for each MPVM session. Below are some guidelines to help you determine the size of PVM server virtual machine you will need:

- The MAXUSER setting times 1200 bytes. The MAXUSER setting defines the maximum number of MPVM users allowed concurrently. The default is 100.
- Each active session times 6-30K bytes. This is the bulk of the storage, used for screen image buffers. The number here depends on terminal sizes, support of extended data streams, and use of program symbol sets. We use 8K per session as an average.

  The default maximum number of sessions is 500: 100 users, 5 sessions per user.
- The number of defined links (PVM and PVMG) times 12K

- The BLOCKS setting times 16K. PVM initially allocates 50 storage blocks (16K each) for the PVM module. However, you might need to increase this value.

Here is an example:

```
  100 users (default) * 1200 bytes per user  =   120K
+ 500 sessions * 8K (average) per session    =  4000K
+ 4 links * 12K (approximate) per link       =    48K
+ 50 blocks (default) * 16K bytes per block  =   800K
                                               ------
                                               4968K
```

Because 1024K = 1M, 4968K is approximately 4.7M. The PVM server virtual machine in this example could probably run comfortably with 5MB of virtual storage.

In the sample directory distributed with PVM, the PVM server virtual machine is defined as 5MB with a maximum of 8MB.

How Much Storage is Need for Your PVMG Server?

The PVMG server virtual machine runs as a disconnected server virtual machine. Below are some guidelines to help you determine the size of the PVMG server virtual machine you will need:

- PVMG program storage = 450K
- Additional storage for each link = 10K
- Additional storage for 50 users = 50K
- Additional language repositories = 43K
- System trace table = 16K
- Link trace table for each link = 8K.

In the sample directory distributed for PVMG, the PVMG server virtual machine is defined as 4M with a maximum of 8M.

Performance Considerations

See "Enhancing PVM Performance" on page 199 to find out ways to improve PVM performance. The performance options should be used for peak PVM performance.

Security Considerations

There are a number of possible security exposures that you need to consider when planning your PVM network. This section covers some of the areas you need to take into consideration when you plan your PVM installation.

Notepad Facility

Sensitive data fields, not usually visible on the user's screen (for example, passwords), can be exposed if copied into the PASSTHRU DATA or *nickname* MPVMDATA files by the MPVM or PASSTHRU Notepad facility. It is the user's responsibility to safeguard this file if a local security problem exists.

Session Security

PVM users begin with a PVM session to gain access to their desired target, then start a session with the target system or application. With *some* targets, the sequence for terminating sessions is important. If termination does not occur properly, a security exposure could exist.

In such cases, it is essential that the target session be terminated first, *before* the PVM session is terminated. If this sequence is not followed, the target session can remain active and open to access by subsequent users.

To avoid this situation, users should follow the proper termination sequence.

Users should be aware that *any* condition that causes the PVM session to end before the user has properly logged off from the target system or application can result in the same security exposure. Some such conditions are:

- The user turns off power to the display station.
- The user turns the key in the display station's security lock (optional hardware feature) to the locked position, then returns it to the unlocked position.
- The user terminates the PVM session by entering the Session Terminate string (default ####) or pressing the designated Session Terminate key.
- The user temporarily disconnects from the target session and does not resume the session within the defined or default (20 minutes) time limit.
- The operator (or other authorized user) issues a DROP command that affects the session.
- PVM abends because of either an internal problem or a problem in the VM operating system.
- A hardware problem affects the user's display station or the communication path between the display station and the target system.

  This event is, of course, beyond the user's control. In some sessions, however, the user can take preventive steps to safeguard session security. These are sessions involving another PVM system or PVMG system. (That is, when the target is either a non-PVM system accessible through another PVM or PVMG system, or the target is an application available on another PVM system.)

  The preventive steps consist of establishing an intermediate PVM session (on the other PVM system) before initiating the target session. The user needs a user ID to log on the other PVM system. Once logged on and running CMS, the user issues the PASSTHRU command and then selects:

  - The desired non-PVM target system, or
  - The PVM system to start a local session and then dials the desired target application on that system.

  The intermediate PVM session protects the target session in the event of a problem affecting the communication path between the two PVM systems.

There are several methods an installation can use to minimize or eliminate this security exposure. It can do one or all of the following:

- Remind users to end *all* target sessions *first* (using the standard logoff procedure). (You can do this by sending a memorandum to PVM users or by issuing a message to them at their display stations.)

- Modify the target application program to require a new logon sequence after it receives a power-on reset condition. (PVM notifies the target of an interruption or termination of the PVM session by indicating a power-on reset condition. The security exposure exists because some target applications ignore the condition.)
- For sessions with non-PVM targets, reserve use of specific ports for specific users or devices.

Configuration File PORT Record

If a user's PVM session is interrupted by an unusual condition, having port usage restricted to that user or display station can prevent another user or display station from accidentally gaining access to the session when the unusual condition is corrected. Two parameters on the configuration file PORT record can be used to reserve port usage.

nodeid
: reserves use of the specified port for exclusive use by PVM users accessing the port from the system node specified by this parameter.

userid/devid
: reserves use of the specified port for exclusive use by the user identified by *userid/devid*.

If neither *nodeid* nor *userid/devid* is specified, an asterisk (*) parameter must be specified to indicate that no access restrictions apply to this port. Complications can arise by reserving 327x logical ports for both CMS users and devices. If a CMS user has a port reserved for his/her user ID and is using a 3270 display station whose real address also has a port reserved, then PVM initiates a session with the lowest number available port with a matching ID. For example, the following port records are defined:

```
PORT 2 3278-2 RMTPVM1 371
PORT 8 3278-2 RMTPVM1 USERA
```

If USERA issues the PASSTHRU command from a 3278 Model 2 Display Station at real address 371 to start a session with a 327x emulator node. Using the preceding PORT definitions, PVM assigns that session to port 2. If USERA logs off from the target node before ending the PVM session, the next user of real device 371 could gain access to USERA's active session. Some ways to avoid this problem are:

- Inform users of this possibility and of the necessity to log off from sessions with non-PVM systems prior to terminating the session.
- When reserving ports for both users and devices, reserve the lower number ports for user IDs and the higher number ports for real device addresses. This assures that CMS users will get ports reserved for their user IDs and not those reserved for specific device addresses.

For printer sessions, however, it is best to specify *nodeid* and *devid*. This ensures that the desired printer is assigned a particular port and prevents output from mistakenly being sent to other printers and other locations. Under unusual circumstances, the selection of a node ID may cause a problem. When a START PRT command is entered, CMS transforms the command parameters into a series of eight-byte tokens. If a keystroke error causes the node ID operand and emulator port number to be concatenated, then CMS treats these values as one parameter and returns the first eight bytes. If the node ID is eight bytes long, then the effect of this operation eliminates the emulator port number. When

CMS returns a START PRT command without an emulator port operand, PVM finds the first unrestricted 3270 printer port and assigns it to the virtual printer. This port may be different from the port that was originally specified on the START PRT command.

To minimize or eliminate this problem you can choose from several options. In assigning emulator node IDs, it can use node IDs with character lengths of less than 8 characters. Then, if a keystroke error connects the node ID and port number, an incorrect node ID error is returned. Another technique includes restricting each logical 3270 port on an emulator node to a specific virtual printer so that the port number is not required on the START PRT command.

Abnormal PVM Termination

If PVM abends, the action taken depends on whether the session is executing in a PVM to PVM (or PVMG) environment or in a 3271/3274 emulator environment.

- PVM to PVM (or PVMG) environment

  If the local PVM or PVMG abends, other PVM or PVMG systems in the network ends :1143essions associated with the abending system. The local VM CP disconnects all users logged on through the local PVM or PVMG and drops all local CP DIAL users.

- 3271/3274 emulation environment

  Following an abend, when the local PVM is reinitialized, all ports on emulator lines are treated as being powered off. When a user establishes a connection with one of these ports, the local PVM emulates a power-on reset condition. It is the responsibility of the remote system to assure that any previous sessions are terminated when it receives the power-on reset. This assures that the next user of a port will not have access to the prior user's session. If the remote installation does not have this capability, the *nodeid* and *userid/devid* parameters of the configuration file PORT records can be used to reserve ports for exclusive usage. See "Configuration File PORT Record" on page 31.

- PVM to remote 4300 processor environment

  Following an abend, when the local PVM system is reinitialized and the ROCF line driver restarted, if the switched line connection is still open, the ROCF line driver breaks the former connection. It does this by simulating the action of hanging-up the telephone receiver.

Restrictions

You should be aware of several PVM restrictions. Some of these restrictions are hardware related, some are system related, and others are implementation restrictions.

General Restrictions

- When logged on to the PVM or PVMG user ID, the operator or administrator should enter these commands to prevent errors:

  ```
  cms set fullscreen off
  cdbc cons off
  ```

  When logged on to the PVMG user ID, the operator must prefix PVMG commands with `pvmg`.

- The TEST REQUEST and SYS REQUEST keys cannot be passed to an application that was accessed by 3271/3274 emulation or from a user's *remote* display station. This is because R3270 control units do not indicate from which display station the request came. Also, the TEST REQUEST and SYS REQUEST keys are not transmitted *to* PVM from either SNA or remote display stations, and therefore, cannot be forwarded by PVM.
- The SYS REQUEST key is not passed when the PASSTHRU command is used.
- The maximum data length that can be transferred as one unit to or from a 3270 display station varies by line type. For APPC, 3088, CTCA, and BSCA link types, the maximum input and output data length is 64K bytes. See corresponding restrictions listed below for 3271/3274 emulator and ROCF lines.
- The PASSTHRU Notepad facility requires that the user's display station have PF keys. MPVM's Notepad facility does not require PF keys. The notepads do not store extended attributes in the PASSTHRU DATA or *nickname* MPVMDATA files. MPVM will store extended attributes if the ATTRIBUTE variable setting is turned on. Also, single APL and TEXT characters whose hex representation requires 2 bytes, such as underscored APL characters, are stored as blanks.
- Issuing the CMS HX command on the PVM user ID while PVM is executing causes unpredictable results. Use the PVM QUIT or SHUTDOWN command to halt execution of PVM. (See Appendix C, "PVM Commands" for information on the format and use of these commands.)
- Printers attached by CP do not support extended data streams.
- The 3270 ERASE ALL UNPROTECTED (EAU) command cannot be totally supported because of a Diagnose X'58' restriction. If an EAU command is received, it is simulated with a WRITE command and the 3270 order, ERASE UNPROTECTED TO ADDRESS. This causes the following conditions:
  - All Modified Data Tags (MDTs) are reset, unlike the EAU command which only resets the MDTs of unprotected fields
  - Always unlocks the keyboard.
- To use extended data streams, both the local node and the target node must be PVM nodes. Both of these nodes must be at Release Level 2 with the Remote 3270 Feature installed, or at a higher release level.

When you use the PASSTHRU command, the local CP takes control of the display station to write any messages. The currently displayed screen is lost and any subsequent application-generated output is displayed on a new screen. This new screen does not have any of the defined fields or data that were present on the previous screen. This makes it impossible to input any data or commands. (By pressing CLEAR, you signal the host application that the screen has been cleared.) This forces the application to rewrite the entire screen. CP TERMINAL BREAKIN GUESTCTL reduces this problem to a minimum but it is available only to locally attached, non-SNA 3270 devices.

PSPI

Access Security Exits Restrictions

The Access Security Exits section of this book provides sample exit code for the PVM server component of PVM. These security exit routines enable an installation to control sessions originating at a local node, outside the local node, and commands that are destined for or originated at another node. The security exits are built in to the PVM module supplied on the PVM 2.1.1 product tape and when built by VMSES/E. If these exits are not required for PVM 2.1.1, place a dummy REMOTE AUTH file on the PVM's 191 disk or a disk which the PVM user id has access to.

PSPI end

GUPI

PCCF Restrictions

- When developing an application using IUCV, the application programmer must consider that the USERDTA field of the IPARML control block is used by both the PVM IUCV application-to-application support and the CMS IUCV support.
- The maximum data length on an IUCV SEND or an IUCV REPLY is 32768 bytes if both the origin and target applications are on VM systems.
- CP IUCV PRIORITY cannot be used unless authorized by appropriate CP Directory statements.
- One-way SENDs are not synchronized and the *message complete* interrupt does **not** mean that the opposite end has received the data.

GUPI end

Channel to Channel Adapter (CTCA) Restrictions

- You cannot have two CTCA links connecting two VM systems through the same two channels.

3271/3274 Emulator Restrictions

- The maximum data length that can be transferred as one unit to or from a 3270 display station is 8080 bytes.
- If the target node issues the READ/BUFFER command, a data string of X'604040' is returned.
- The READ/MODIFIED command is supported. However, if there is no immediate data to send when the target node issues this command, the data string X'604040' is returned.
- The data string X'604040' is sent if the target node attempts to reread data already sent by a previous POLL or READ/MODIFIED command. This may occur when the target node uses a small input buffer intending to reread into a larger buffer if the input record is too large for the initial buffer. To avoid this problem the initial buffer and data read count should handle the maximum record size.
- Incorrect 3270 orders do not result in an error status being sent to the host because the orders are not checked by the emulator. They will, however, cause the user session to stop when the orders are written to the user's 3270 display station.

- The COPY function (from 3270 Information Display System printers) is not supported.
- If a 370X TP controller is used, it must have at least one Type 2 communication scanner that supports the 2700 tributary feature.
- The use of extended data streams is not supported for the emulator line driver.

Logical Device Support Restrictions

- All command-chained CCWs are treated as separate unchained CCWs. Therefore, results that depend on commands being chained are unpredictable. The one emulated condition is a READ chained from a WRITE of length 4, containing a WCC character and a SBA character. A scan routine processes the data from the READ and discards all data prior to the address referenced by the SBA in the WRITE command. Note though that this emulated condition does not occur if the CCW chain is issued to a virtual display station. In this case, CP ignores the WRITE CCW.
- Data-chained CCWs have the data combined into a single buffer. The combined length cannot exceed the PVM data length restrictions.

3270 Information Display System Printer Restrictions

- Using 327x emulation, PVM supports printer sessions between CP owned local or remote 3270 Information Display System printers and non-PVM target systems. You can assign any 327x logical port to a directly attached 3270 Information Display System printer. Printer ports are defined in the configuration file using the PORT record. PVM lets you to dedicate a printer at your local system to producing output for users working on target applications. You attach printers to the local system using the CP ATTACH (or CP NET ATTACH) command or by a DEDICATE statement in the VM directory for the PVM user ID. The target applications must support the 3270 Information Display System printers.
- The printer sessions described above are **not** the same as sessions with printers attached as part of remote clusters managed by PVM. If you want to access a remote printer managed by PVM, you must specify the proper information on the CLPORT configuration file record.

  To attach a remote printer to a PVM system, use the following form of the CLPORT record:

  CLPORT *paddr devtype nodeid addr vmid vaddr*

  This:

  - Defines a printer
  - Allows a logical device representing the printer to be created on a CP system (*nodeid*)
  - Assigns an offset value for the logical device address (*addr*), or if the *addr* is set to zero, the next available logical printer device address will be assigned by PVM
  - Causes PVM to attach the logical device to a virtual machine (*vmid*) at a given address (*vaddr*).

To attach a remote printer to a non-PVM node, use this form of the CLPORT record:

CLPORT *paddr devtype nodeid addr*

This defines a printer and causes connection to a specific port (*addr*) on an emulator link (non-PVM *nodeid*). If a port number is not specified, the connection is made to the next available port.

If you have properly specified the CLPORT record for the remote printer attachment you want, PVM attempts to start a printer session when it starts the line.

Note: Issuing the DROP user command on a printer session will cause the logical session to be terminated. PVM will attempt to start a new printer session.

ROCF Restrictions

- The maximum data length that can be transferred as one unit to or from a 3270 display station is 4040 bytes.
- Only local users (those logged on to the local system or accessing from a display station connected to the local system) can access those ROCF nodes defined on the local PVM system.
- The TEST REQUEST and SYS REQUEST keys are not passed by these remote devices to PVM. These keys are used to change program function mode settings on models of the 3277 and 3278 display stations that have only 12 PF keys. Without this change to System Function ([S]) mode, these devices cannot serve as ROCF consoles. Therefore, remote 3277 and 3278 display stations used for such sessions must be equipped with 24 PF keys.
- Only one session can be active for each ROCF node.
- The use of extended data streams is not supported by the ROCF line driver.

Users should also refer to the operator's guide pertinent to the remote 4300 processor they will be operating to determine procedures and restrictions for use of remote consoles.

Migrating from PVM 1.4

If you are migrating from PVM 1.4 to Version 2 of PVM, you should maintain a backup of you current PVM 1.4 installation during migration.

PVM Version 2 will replace all existing PVM 1.4 files.

PVM Version 2 configuration file support is compatible with existing PVM 1.4 functions. New records have been added to support the MPVM and PVMG components of PVM Version 2.

The PVM LINK configuration file record has changed to support the new IUCV line driver for communications with the PVMG component.

There are several new PVM commands that provide support for the new functions and components of PVM Version 2.

If you plan to use the MPVM component of PVM Version 2, you should set up the MPVMSYS NAMES file to describe commonly accessible nodes and SNA applications for your MPVM users.

If you plan to install PVM's PVMG component, you will need to set up a PVMG configuration file, and plan the PVMG server's role in your PVM and SNA networks.

Using PVM Version 2 in a VM/ESA or VM/XA Environment

VM/ESA and VM/XA support PVM as a networking and distributed processing tool. This section describes the differences between using PVM with VM/ESA or VM/XA, and using PVM with VM/ESA (370 Feature) or VM/SP. For more information on the VM/ESA CP commands see the *VM/ESA CP Command and Utility Reference SC24-5519*. For more information on the VM/XA CP commands see the, *VM/XA SP:CP Command Reference*.

The following are restrictions and considerations which apply to PVM operations under VM/ESA and VM/XA. **All other considerations, requirements, and restrictions described earlier in this chapter also apply to operating PVM in a VM/ESA or VM/XA environment**, unless otherwise noted.

Virtual Machine Mode

The PASSTHRU command can be processed from a 370, XA, XC, or ESA mode virtual machine. When the virtual machine mode is XA or XC, PASSTHRU runs in XA-toleration mode below the 16MB line. The PVM MODULE runs in a 370 virtual machine only. The MPVM and PVMG components are enabled for XA, XC, 370, and ESA modes. PVMG executes in an XA, XC, or ESA virtual machine below the 16 megabyte address line.

Line Drivers

You cannot use the APPC link type on a VM/XA system.

PVM TRACE Command

VM/ESA and VM/XA do not support the PVM TRACE command with the SYSTEM ETRACE options.

Virtual Machine Dumps

The Interactive Problem Control System (IPCS), a component of VM/ESA (370 Feature) and VM/SP, and the Dump Viewing Facility, a component of VM/ESA and VM/XA, support analyzing virtual machine dumps. If you want to analyze a dump using either of these facilities, specify VMDUMP and the virtual machine ID of the virtual machine to receive dumps on the DUMP configuration file record.

CP Remote Support

The Control Programs (CP) of VM/XA do not contain any remote BSC support. This means that there are no equivalents to the VM/ESA (370 Feature) or VM/SP NETWORK command, remote DIAL, or 37XX Emulation Program (EP) load or dump. (37XX refers to the IBM 3704, 3705, and 3725 Communications Controllers.) You cannot use remote printers. You must load a 37XX using an operating system other than VM/XA.

Hardware

You can only use hardware that is supported by VM/ESA and VM/XA. Not all of the hardware mentioned in this book is supported by those systems.

Performance

The performance options available in a VM/ESA (370 Feature) or VM/ESA environment are:

- Reserved pages using the CP SET RESERVE command
- Locked pages using the CP LOCK command.

The SET RESERVE command is for class A privileged users. The PVM operator needs class A privilege to enter the command.

You can follow the performance recommendations listed for ROCF sessions with VM/XA.

Requirements

The following requirements apply to VM/ESA and VM/XA operations:

- You **must** install PVM service to the level indicated in the VM/XA Program Directory. This level allows PVM to work with VM/XA. After that, you may install additional service as needed.
- If you are using a 37XX, you must load its emulation program using an operating system other than VM/XA. (VM/XA cannot load a 37XX.)

Installing PVM 2.1 on a VM/ESA or VM/XA System

Note: See the Program Directory for installing PVM 2.1.1.

When you install PVM on VM/ESA or VM/XA you:

1. Install PVM from its product tapes
2. Apply service to the PUT level described in the VM/ESA or VM/XA Program Directory
3. Install any additional service.

Consider these points when installing PVM on VM/ESA or VM/XA:

- Follow the standalone method for installing PVM as described in "Loading Files - The Standalone Method" on page 48. You cannot use the quick method because VM/XA does not have the INSTFPP exec.
- Make sure your installation uses an operating system other than VM/XA to load the 37XX emulation program. (VM/XA cannot load a 37XX.)
- The IBM-supplied VM/ESA or VM/XA system directories allocate the minidisks that you need to install PVM. The directory also contains links to the PVM disks for the MAINT user ID.

  The major difference between VM/XA and other VM system directories is the VM/XA does *not* support DEDICATE NETWORK statements.
- Assign class G and one other privilege class to PVM. Class G lets the PVM virtual machine IPL CMS (190). PVM may need a different class to issue Diagnose X'7C' when creating logical devices, depending upon the VM system you are using. (In VM/ESA, Diagnose X'7C' is available to users with a class privilege of A, B, C, D, E, or F.) If you choose class B, PVM can also

attach telecommunication lines by issuing ATTACH commands using real addresses from PVM user ID.

- You **must** install service to the level indicated in the VM/ESA or VM/XA Program Directory. This level allows PVM to work VM/ESA or VM/XA. After that, install additional service as needed.
- A PVM CNTRL file defines the MACLIB libraries that PVM uses. These libraries are different for VM/ESA and VM/XA installations. The PVM CNTRL file can be found on PVM's base TEXT staging minidisk (MAINT's 49E). To update PVM CNTRL for a VM/ESA or VM/XA installation:
  - Edit the file.
  - Type an asterisk (*) at the beginning of the two lines you do not want to use. This *comments out* the unwanted macro libraries.
  - Type in an uppercase *T* in the first column of the second line. The second line contains the MACLIB libraries for VM/ESA installations. (The sixth line contains the MACLIB libraries for VM/XA. The eighth line contains them for VM/SP).
  - Place the updated file on the base TEXT minidisk (MAINT's 49E).

Storage Considerations

To calculate an approximate value for PVM storage requirements, use the values from the configuration file along with certain fixed storage requirements. Use the result on the USER statement of the system directory. The VM/ESA or VM/XA machine requires 4MB for the CMS low storage size.

ESCON Channel Support

The PVM ESCON I/O interface provides for attachment of control units or I/O devices to ESA/370* channel subsystems through a serial channel. The PVM ESCON and parallel I/O interfaces have the same basic functions for commands, status and data. The means for transmitting these functions on the two interfaces are very different.

The transmission medium for the ESCON I/O is called a link, and consists of two conductors, one for receiving and one for sending. The ESCON channel provides much higher data rates and greater transmission distances than the parallel channel.

PVM supports connections through:

- ESCON channel using a 3088 Model 1F Control Unit to another PVM server virtual machine
- Terminals and printers attached to an IBM 3174 Control Unit models *12L* or *22L*.

Chapter 4. Installing PVM 2.1

Note: For installing PVM 2.1.1, see the Program Directory supplied with the product tape.

This chapter tells you how to install PVM on your VM system. The procedure consists of the required steps that follow. These steps assume that you will install PVM from the MAINT (or installer's) user ID.

1. Add directories for the PVM and PVMG operators
2. Setup the MAINT (or installer's) user ID
3. Get disks for PVM, MPVM, and PVMG files
4. Load the tape files to the disks
5. Set up autolog statements for PVM and PVMG
6. Setting Up the PVMG Server Virtual Machine
7. Add PVMG to GCS
8. Add VTAM definitions
9. Apply service
10. Verify the initial installation

After you install the files from tape, you must also perform the following tasks to make PVM operational:

1. Define your network by setting up a PVM and PVMG configuration files (described in Chapter 6, "Defining Your PVM Configuration" and Chapter 7, "Defining Your PVMG Configuration")
2. Create the MPVMSYS NAMES file
3. Apply preventive service for your system if such service is available (described in Chapter 11, "Applying Service to PVM 2.1")
4. Optionally customize your PVM system by:
   a. Installing the PVM National Language Feature (described in Chapter 14, "National Languages on Your PVM System")
   b. Developing exits (described in "Developing Exits" on page 209).

Step 1. Add Directory Entries for the PVM and PVMG Operators

The administrator and operator use a system MAINT user ID to install PVM, and optionally, PVMG. A PVM user ID is used to set up and define the local PVM system and the PVM network.

If you are installing PVMG, you will need to set up a PVMG user ID and use it to set up and define the local PVMG system and its connection to the PVM network. The PVMG server virtual machine runs in the Group Control System (GCS) and provides the gateway between PVM and VTAM for sessions to an SNA network. This virtual machine is optional. If you do not intend to establish sessions between PVM and the SNA network, you do not need to create a directory entry for the PVMG server virtual machine. From the PVM or PVMG user ID, an administrator or operator can enter any PVM or PVMG command. Also, error messages that are the result of conditions that need system programmer action are sent to the PVM or PVMG user ID.

You can set up the PVM and PVMG user IDs in your VM system directory. For more information on the directory statements, see the *VM/ESA: Planning and*

© Copyright IBM Corp. 1991,1993

Administration, SC24-5521 and *VM/ESA: CP Planning and Administration for 370, SC24-5430.*

Note: After you create source directory entries for these virtual machines, ensure that you apply them to the VM system's online directory of virtual machines. For more information about VM's directory, see the following books:

- *VM/ESA: Planning and Administration, SC24-5521*
- *VM/ESA: CP Planning and Administration for 370, SC24-5430*
- *VM/ESA Connectivity Planning, Administration, and Operation, SC24-5448*

PVM needs two minidisks:

- 191, to contain a PROFILE EXEC;
- 199, to contain common files for all PVM users. This disk is known as the MPVM common disk. All PVM users need read-only access to this disk and must be allowed to link to it.

Sample PVM Directory

Here is what a directory entry for PVM would look like:

Table 4 (Page 1 of 2). PVM User ID System Directory

Directory Statement	Comments
USER PVM XXXXXXXX 5M 8M BG 50	Class B privilege must be assigned to PVM if: • The configuration file (PVM CONFIG) for that machine is to contain a MSGNOH record, or • to allow warning messages to be issued to PVM users, or • The configuration file contains CLPORT records that specify target virtual machine identifications for remote printers. When PVM is started, it automatically issues an ATTACH to the virtual machine using the logical device address specified on the CLPORT record, or • Telecommunication lines and local 3270 printers are attached by ATTACH statements in a PROFILE EXEC or issued from the PVM console. For remote 3270 printers not defined by the PVM CONFIG file, use NET ATTACH. PVM storage size depends on the needs of an installation. The number of links defined in PVM CONFIG file, or the number of users, add to the amount of storage you need. The recommended minimum storage size for PVM is 5MB. See "How Much Storage Is Needed for Your PVM Server?" on page 28.
OPTION ECMODE BMX DIAG98 MAXCONN 4096 VCUNOSHR	The DIAG98 and VCUNOSHR options are not available on VM/XA or VM/ESA systems. The ECMODE and BMX options are not supported in VM/ESA. ECMODE is handled by the CP SET MACH command.
IUCV ANY M 255 IUCV ALLOW IUCV *IDENT	Add these IUCV directory statements to effectively use IUCV communication (for APPC, IUCV,and PCCF type links, or for MPVM user tasks). This minimizes problems where source or target virtual machines do not have appropriate IUCV statements.
IPL CMS	
CONSOLE 009 3215	The CONSOLE address must be specified as shown.
SPOOL 00C 2540 READER A SPOOL 00D 2540 PUNCH A SPOOL 00E 1403 A	The SPOOL addresses should be specified as shown.

Table 4 (Page 2 of 2). PVM User ID System Directory	
Directory Statement	**Comments**
LINK MAINT 36E 191 MR	Suggested disk use
LINK MAINT 38F 199 RR	Suggested disk use
LINK MAINT 190 190 RR	Suggested disk use
LINK MAINT 19A 19A RR	Suggested disk use
LINK MAINT 19B 19B RR	Suggested disk use
LINK MAINT 19C 19C RR	Suggested disk use
LINK MAINT 19E 19E RR	Suggested disk use
LINK MAINT 19F 19F RR	Suggested disk use

The MAINT user ID owns the 190, 19D, 193, 19E, and 36E minidisks. These are all disks that can be used during installation of PVM. MAINT owns the 36E (PVM System minidisk) which PVM links to as 191.

For performance reasons OPTION REALTIMER should not be specified. In addition, consideration should be given to specifying a better dispatching priority for PVM than the default (64). For example, a priority of 50 is higher than the default priority. The dispatching priority is specified on the USER directory control statement.

For directory and installation considerations with VM/ESA and VM/XA, see "Using PVM Version 2 in a VM/ESA or VM/XA Environment" on page 37. For general information on updating VM/XA and VM/ESA directories, see the following books:

- *VM/ESA: Installation for 370, SC24-5431*
- *VM/ESA: Planning and Administration, SC24-5521*
- *VM/ESA: CP Planning and Administration for 370, SC24-5430*
- *VM/XA: System Product Installation and Service, SC23-0364*

Sample PVMG Directory

Here is what a directory entry for PVMG would look like:

Table 5 (Page 1 of 2). PVMG User ID System Directory	
Directory Statement	**Comments**
USER PVMG XXXXXXXX 4M 8M BG 49	Class B privilege must be assigned to PVMG if: • The configuration file (PVMG CONFIG) for that machine is to contain a MSGNOH record. PVMG storage size depends on the needs of an installation. The number of links defined in PVMG CONFIG or the number of users add to the amount of storage you need. The recommended minimum storage size for PVMG is 4MB.
IPL GCS	
IUCV ANY	Allows PVMG to effectively use IUCV for communications.
OPTION ECMODE	Option ECMODE is not supported by VM/ESA. ECMODE is handled by the CP SET MACH command.
CONSOLE 009 3215	The CONSOLE address must be specified as shown.
SPOOL 00C 2540 READER A SPOOL 00D 2540 PUNCH A SPOOL 00E 1403 A	The SPOOL addresses should be specified as shown.

Table 5 (Page 2 of 2). PVMG User ID System Directory	
Directory Statement	**Comments**
LINK MAINT 38E 191 MR	Suggested disk use
LINK MAINT 595 595 RR	Suggested disk use
LINK MAINT 190 190 RR	Suggested disk use
LINK MAINT 19D 19D RR	Suggested disk use
LINK MAINT 19E 19E RR	Suggested disk use

Step 2. The MAINT (or Installer's) User ID

The PVM and PVMG user IDs link to the system MAINT user ID. Installation and service of licensed programs is often done on the MAINT user ID. Installation and service can, however, be performed from another user ID if you choose. PVM and PVMG share a common set of minidisks for these activities along with other licensed programs. Here is how the MAINT directory entry appears:

```
USER MAINT XXXXX 16M 16M ABCDEFG
ACCOUNT 1 SYSPROG
OPTION
IPL 190
CONSOLE 009 3215
SPOOL 00C 2540 READER *
SPOOL 00D 2540 PUNCH  A
SPOOL 00E 1403 A
MDISK 19D 3380 nnn 005 VMPK01 MW ALL      WMAINT   MMAINT
MDISK 19E 3380 nnn 002 VMPK01 MR RMAINT   WMAINT   MMAINT
MDISK 29E 3380 nnn 007 VMPK01 MR RMAINT   WMAINT   MMAINT
MDISK 39E 3380 nnn 032 VMPK01 MR RMAINT   WMAINT   MMAINT
MDISK 49E 3380 nnn 014 VMPK01 MR RMAINT   WMAINT   MMAINT
MDISK 36E 3380 nnn 006 VMPK01 RR RMAINT   WMAINT   MMAINT
MDISK 38E 3380 nnn 002 VMPK01 RR RMAINT   WMAINT   MMAINT
MDISK 38F 3380 nnn 005 VMPK01 MR RMAINT   WMAINT   MMAINT
MDISK 298 3380 nnn 001 VMPL01 MR RMAINT   WMAINT   MMAINT
```

Because MAINT owns the 36E and 38E minidisks in RR mode, PVM and PVMG can link to the disks in MR mode. Note that *nnn* is an installation-supplied address.

Step 3. Getting Disks for PVM, PVMG, and MPVM Files

Use the MAINT user ID for installing PVM and PVMG. All of the PVM files are not loaded onto a single minidisk. Instead, groups of files are loaded onto several of MAINT's minidisks. Lists of the tape files that you will be loading onto the disks can be found in Appendix B, "Tape Layouts for PVM 2.1."

The minidisk addresses that are referred to in this book are suggested addresses. You are free to choose the addresses that you want. If you choose to use other addresses, remember to consistently use your addresses in place of the suggested ones. However, you should use the suggested addresses if you install PVM by the quick method described on page 48. Examples in this book use the suggested addresses.

The MAINT minidisks that you will be using are:

Minidisk	MAINT's Address	Comments
Installation exec, softcopy publications, and sample directory files	191	Mandatory disk. This is the minidisk containing the product identifier, the installation exec, softcopy PVM publications, memo to users, and the PVM and PVMG sample directory files. These are tape files 1, 2, and 3.
PVM system	36E (PVM's 191)	Mandatory disk. The PVM system minidisk is the A-disk for PVM, address 191. We suggest that this also be MAINT's 36E. (The PVM user ID has a multiread link to MAINT's 36E.) It should contain the contents of distribution tape file number 4.
Service Staging	29E	Mandatory Disk. Contains the updated assemble, copy, macro, and repository files for service.
System extension	19E	Mandatory disk. This is a minidisk to which users have common read access (such as, the Y-disk extension to the CMS system S-disk). This should be MAINT's 19E. It should contain the contents of distribution tape file number 5.
Source staging	39E	Optional disk. The source staging minidisk is used for the optional PVM and PVM/IPCS source ASSEMBLE, COPY, and MACRO files. This should be MAINT's 39E disk. It should contain the contents of distribution tape file number 6.
Base TEXT staging, Service/Build Execs	49E	Optional disk. The base text staging minidisk should be the MAINT user ID's 49E disk. It should contain the contents of distribution tape file number 7.
PVM VM/ESA HELP files	19D	Optional disk. The CMS minidisk containing HELP files to which users have common access. **Note**: Help for PVM messages with identifiers is only available in PVM's VM/ESA HELP files. If you need help for these messages, PVM help files must be installed. This should be MAINT's 19D disk. The PVM HELP files should be added on this minidisk. The HELP files are loaded from distribution tape file number 8.
IPCS text	193	Optional disk. MAINT's 193 minidisk can also be used for the PVM/IPCS TEXT and IPCS HELP files. These files can be found on distribution tape file number 9. Do not load these files if your installation runs VM/ESA or VM/XA as the base system.
PVMG System Files	38E (PVMG's 191)	Optional Disk. This optional minidisk contains PVMG system files and the sample PVMG PROFILE EXEC for PVM's PVMG optional component. This is tape file number 10.
MPVM system files	38F (PVM's 199)	Optional Disk. This disk contains the MPVM system files for PVM's MPVM optional component. This is tape file number 11.
Sample VTAM definition file	298 (VTAM's 191)	Optional disk. The VTAM definition minidisk contains the sample VTAM definition file for PVMG. It is tape file 12. This should be MAINT's 298 disk.

The PVM system minidisk and a system extension minidisk (with load modules and execs) are needed to install and run PVM. The system extension disk (19E) holds the load modules and execs. The IPCS disk (193) holds the IPCS TEXT files. The IPCS TEXT files are optional. The 49E and the 39E minidisks are also optional. The service staging minidisk (29E) is needed only if you plan to apply service.

After PVM is installed, the PVM user ID contains the files that are used by PVM. These files include:

Filename	Filetype	Comments
PVM	MODULE	Used by the PVM server virtual machine.
PVM	MAP	Shows where TEXT decks are loaded.
PVM	PRELMAP	Created prior to the PVM MAP.
RUNPVM	EXEC	Starts PVM.
PROFILE	EXEC	Contains system (CP) commands (such as the CP ATTACH and NETWORK ATTACH commands). PROFILE EXEC is used when you are automating PVM system initialization. A sample PROFILE EXEC is provided with the distribution tape that you can update for your installation.
PROFILE	PVM	Contains PVM commands. PROFILE PVM is used when you are automating PVM system initialization. A sample PROFILE PVM is provided with the distribution tape that you can update for your installation.
PVM	CONFIG	Contains PVM configuration file records. PVM CONFIG defines PVM to the system. A sample PVM CONFIG is provided with the distribution tape that you can update for your installation.

After PVMG is installed, the PVMG user ID contains the files that are used by PVMG. These files include:

Filename	Filetype	Comments
PVMG	LOADLIB	Used by the PVMG server virtual machine.
PVMG	LKEDIT	Shows where TEXT decks are loaded.
RUNPVMG	EXEC	Starts PVMG.
PROFILE	EXEC	PROFILE EXEC is used when you are automating PVMG system initialization. A sample PROFILE EXEC is provided with the distribution tape that you can update for your installation.
PROFILE	GCS	Contains PVMG commands. PROFILE PVMG is used when you are automating PVMG system initialization. A sample PROFILE PVMG is provided with the distribution tape that you can update for your installation.
PVMG	CONFIG	Contains PVMG configuration file records PVMG CONFIG defines PVMG to the system. A sample PVMG CONFIG is provided with the distribution tape that you can update for your installation.

Step 4. Load the Tape Files to Disks

The first real, logical tape file you load contains sample directory entries for the PVM server virtual machine (PVM DIRECT) and PVMG server virtual machine (PVMG DIRECT).

The following table shows the layout of the PVM product tape.

Note: The tape is divided into sections called logical tape files. Each logical tape file contains one or more real files that will get dumped to a certain minidisk. Logical tape files are separated by a single tape mark, and there are two tape marks at the end of tape.

Logical Tape File Number	Target User ID	Minidisk Address	Contents of the Logical Tape File
1	MAINT	191	Product identifier, install exec, softcopy publications
2	MAINT	191	Memo to Users
3	MAINT	191	Sample PVM and PVMG directory files
4	MAINT	36E	PVM system file (PVM module, config files, and loadlib)
5	MAINT	19E	System extension files (DVMUSI module, PASSTHRU and MPVM commands)
6	MAINT	39E	Product source files (Assemble, Copy, and Macros)
7	MAINT	49E	Product base text files (TEXT, MACLIB, CNTRL, and service/build execs)
8	MAINT	19D	PVM help files
9	MAINT	193	IPCS help and text files
10	MAINT	38E	PVMG system files (PROFILE exec)
11	MAINT	38F	MPVM system files (MPVM user module)
12	MAINT	298	Sample VTAM definition file

PVM can be installed from the distribution tape in two ways.

- A quick method using an automatic installation exec, INSTFPP, which calls the PVM installation exec. INSTFPP is an installation aid that comes as part of the VM/ESA base. The PVM installation exec automatically places the PVM files on the minidisks listed previously. (Note, the PVM installation exec formats any unformatted minidisks.) Not all users have the INSTFPP exec. For example, if you are running VM/XA, you do not have INSTFPP.

  INSTFPP expects minidisks to have the addresses suggested in this chapter. INSTFPP uses temporary disk space; make sure that your system has enough temporary disk space available for use by INSTFPP before using the exec.

- A standalone method where you load the tape files rather than an exec. Using the VMFPLC2 command, you can get to the necessary product base and source files needed for your system. This procedure assumes that new minidisks have already been formatted by the CMS FORMAT command.

  If you have not been using the suggested minidisk addresses, substitute yours for the ones used in the example.

Note:

- When using 3380 type DASD, format the minidisks in a 4K block size. For other DASD types, format the minidisks in a 2K block size.

Loading Files - The Quick Method (Using INSTFPP)

1. Log on MAINT user ID, define storage, and IPL CMS. If you are not already logged on as MAINT, issue the following commands:

   ```
   logon maint
   (Enter password)
   def stor 16m
   ipl cms
   ```

2. Mount the PVM distribution tape and attach the tape drive to MAINT as 181. Issue the following commands:

   ```
   cp attach raddr MAINT 181
   cp rewind 181
   ```

   Substitute the tape drive address for *raddr*.

3. Invoke the INSTFPP EXEC with the Product Identification Number. Issue the following command:

   ```
   instfpp 5684100
   ```

   After invoking the exec, answer any questions INSTFPP displays. INSTFPP then loads the PVM installation exec from the tape and prints the memo-to-users which gives additional information about installing and initializing PVM.

   INSTFPP then starts the PVM installation exec. Answer any questions the installation exec displays. The installation exec completes the installation of PVM.

   After the installation of PVM is complete, INSTFPP asks you if you want to install the PVM national language feature. You should **not** install the feature tape now unless your system meets the requirements for the installation of the tape. Read the section "When to Install the National Language Feature" on page 50 for more information.

Loading Files - The Standalone Method

1. Log on MAINT user ID, define storage, and IPL CMS. If you are not already logged on as MAINT, enter the following commands:

   ```
   logon maint
   (Enter password)
   def stor 16m
   ipl cms
   ```

2. Mount the PVM distribution tape and attach the tape drive to MAINT as 181. Issue the following commands:

   ```
   cp attach raddr maint 181
   cp rewind 181
   ```

   substitute the tape drive address for *raddr*.

3. To locate the correct position for PVM Version 2 base files enter:

   ```
   vmfplc2 scan I5684100 MEMO (EOT
   ```

4. Move to the start of the PVM Version 2 system files by entering:

   ```
   vmfplc2 fsf 1
   ```

5. To finish loading the PVM product tape, enter the following commands to load tape file #3 onto the MAINT 191 disk:

   ```
   access 191 a
   vmfplc2 load * * a
   ```

   This tape file contains sample directories and must be loaded. 191 is the address of those sample directories.

   Access the following disks:

   - MAINT 191
   - MAINT 19D
   - MAINT 19E
   - MAINT 36E
   - MAINT 38F
   - MAINT 39E
   - MAINT 38E
   - MAINT 49E
   - MAINT 298.

   You will be prompted for the MR passwords when linking to these disks.

6. Place tape file #4 (PVM system files) on MAINT 36E:

   ```
   access 36E a
   vmfplc2 load * * a
   ```

7. Place tape file #5 (System Extension files) on MAINT 19E:

   ```
   access 19E a
   vmfplc2 load * * a
   ```

8. Place tape file #6 (PVM source files) on MAINT 39E:

   ```
   access 39E a
   vmfplc2 load * * a
   ```

   Or, if you wish to bypass loading of these files:

   ```
   vmfplc2 fsf 1
   ```

9. Place tape file #7 (PVM base text files) on MAINT 49E:

   ```
   access 49E a
   vmfplc2 load * * a
   ```

   Or, if you wish to bypass loading of these files:

   ```
   vmfplc2 fsf 1
   ```

10. Place tape file #8 (PVM help files) on MAINT 19D:

    Note: Numbered messages for PVM, MPVM, and PVMG are not contained in hardcopy. If you wish to view these messages, you ***must*** load the help files.

    ```
    access 19D a
    vmfplc2 load * * a
    ```

    Or, if you wish to bypass loading of these files:

    ```
    VMFPLC2 FSF 1
    ```

11. Place tape file #9 (IPCS help and text files) on MAINT 193:

    ```
    access 193 a
    vmfplc2 load * * a
    ```

    Or, if you wish to bypass loading of these files:

    ```
    vmfplc2 fsf 1
    ```

    If you are not installing PVMG and MPVM, skip to step 15.

12. Tape file #10 contains PVMG files. Get the PVMG files on 38E:

    ```
    access 38E a
    vmfplc2 load * * a
    ```

    If you *do not* want to install PVMG, enter the following command to bypass loading these files:

    ```
    vmfplc2 fsf 1
    ```

13. Tape filel #11 contains the MPVM System files.

    Place tape file #11 on the MAINT 38F disk by issuing the following

    ```
    access 38F a
    vmfplc2 load * * a
    ```

    Or, if you are *not* installing MPVM, enter this command to bypass loading these files:

    ```
    vmfplc2 fsf 1
    ```

14. Place tape file #12 on the MAINT 298 disk (VTAM files for PVMG) by issuing:

    ```
    access 298 a
    vmfplc2 load * * a
    ```

    (VTAM links to the MAINT 298 disk as its 191 disk.)

    Or if you are not installing PVMG,

15. Rewind and detach the tape:

    ```
    cp rewind 181
    cp detach 181
    ```

16. Get MAINT's A-disk back:

    ```
    access 191 a
    ```

When to Install the National Language Feature

PVM is shipped with mixed-case American English and uppercase American English language files. The sample configuration file defines mixed-case American English as the default national language. You can use the LANG configuration file record to automatically set the system national language. PVM users receive PVM messages, menus, and HELP panels in the system national language.

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

You install the national language feature tape files **after** you install the PVM distribution tape. Chapter 14, "National Languages on Your PVM System" describes how to install the national language feature tape.

Step 5. Set Up Autolog Statements for PVM and PVMG

The PVM and PVMG server virtual machines, as well as VTAM and GCS, must be automatically logged on with each system IPL. The AUTOLOG1 virtual machine handles automatic logging on of virtual machines, according to statements in its PROFILE EXEC file. On your system, the file containing the autolog statements may be different.

So, you must update the PROFILE EXEC on the AUTOLOG1 user ID's filemode A minidisk. Here are the steps to do this:

1. If you system does not have RACF installed, you must get the write password for the AUTOLOG user ID's 191 disk.
2. Issue the following command to link to the AUTOLOG1 virtual machine:

   ```
   LINK AUTOLOG1 191 maintaddr W
   ```

   where *maintaddr* is an available address for MAINT. To complete this link, you need the write password to the AUTOLOG1 191 disk that you got in step 1.
3. Issue the following command to access the disk:

   ```
   ACCESS maintaddr A
   ```
4. Make sure CP AUTOLOG statements for the necessary virtual machines appear in AUTOLOG1's PROFILE EXEC in this order:

   ```
   :
   CP AUTOLOG GCS pswd
   :
   CP AUTOLOG VTAM pswd
   :
   CP AUTOLOG PVM pswd
   :
   CP AUTOLOG PVMG pswd
   :
   ```

   We show gaps in between these autolog statements, because you should evenly space them, to give each one enough time to initialize before the next is brought up.

   Note that the user IDs for these service virtual machines may be different on your system.
5. Issue the following command to release the disk:

   ```
   DETACH maintaddr
   ```

PVM's 191 PROFILE EXEC must call the RUNPVM command as one of its final statements. RUNPVM command loads and processes the PVM MODULE and the PROFILE PVM file, if one is found.

PROFILE PVM must be setup to start the installation-defined LINEs, GRAFs, and PRTs.

PVMG's 191 PROFILE GCS must IPL GCS as one of its final statements. The RUNPVMG command must contain statements that load and processes the PVMG LOADLIB.

Step 6. Set Up the PVMG Server Virtual Machine

If you want to establish PVM sessions through VTAM, you must complete this step to properly set up the PVMG service machine. PVMG runs in GCS, in the same group virtual machine as VTAM. The PVMG program is processed from the PVMG LOADLIB file. This section assumes that the PVMG user ID has been created.

1. IPL CMS
2. Log on PVMG user ID
3. Issue:

   ```
   COPY PVMG CONFSAMP A5 = CONFIG A2
   ```

   This copies the sample configuration file for the PVMG server machine to the name that will be used during PVMG's initialization. This file contains installation-dependent configuration information about the PVMG server. See Chapter 7, "Defining Your PVMG Configuration" on page 85 for information about configuring your PVMG server.
4. Issue:

   ```
   IPL CMS
   ```

   This processes PVMG's PROFILE EXEC. The last statement in PVMG's PROFILE EXEC is

   ```
   EXEC RUNPVMG
   ```

   This processes RUNPVMG, which IPLs GCS to start up.

```
runpvmg
SP5 GCS BUILT 08/17/90
Generated at 09/10/91 15:58:57
GCTACC423I A (0191) R/W
Ready;
09/20/91 18:52:01 EFGMAI040I VM Pass-Through Gateway Services 2.1.0000,
built 09/20/91 16:40:19, is ready
09/20/91 18:52:01 EFGOPE625I The PVMG command processor is ready
```

5. If all is OK, enter: pvmg disc

   Disconnecting leaves the PVMG server waiting to service SNA sessions

Step 7. Add PVMG to GCS

The PVMG server virtual machine must be configured as an authorized member of a VM Group Control System (GCS) group that runs VTAM.

For more information on how to authorize the PVMG user ID as a member of the GCS group that runs VTAM, see the *VM/ESA: Group Control System Reference* and the *VM/ESA Group Control System Reference for 370*.

Step 8. Add VTAM Definitions

The PVMG server virtual machine runs as a VTAM application. The supplied PVMG VTAMLST file, which you installed on the VTAM 191 disk, contains the VTAM network resource definitions for PVMG. You must perform these steps to make sure VTAM is set up properly to work with PVMG:

1. Every time VTAM is initialized, you must vary the PVMG resource definition online.

   There are two ways to do this:

   a. Dynamically, from the VTAM operator's console, by issuing:

      VTAM VARY NET,ACT,ID=PVMG

   b. Or, for automatic initialization of VTAM:

      1) LINK MAINT 298 *maintaddr* W (This step still assumes that VTAM links to the MAINT 298 disk as its 191; you installation may use something different for VTAM's 191.)

      2) ACCESS *maintaddr maintmode*

      3) XEDIT ATCCON*nn* VTAMLST, where *nn* is an installation-dependent number

      4) Add PVMG to the list of existing resource names

      5) FILE

      where *maintaddr* is the address assigned to the MAINT virtual machine, and

      maintmode is any free file mode assigned to MAINT's virtual machine.

      The next time VTAM is initialized, it reads this file and will automatically VARY on the PVMG resource.

2. Examine the PVMG VTAMLIST file.

   This file contains VTAM APPL statements that define resource to VTAM:

   - Terminals are allowed to use PVMG. VTAM sees these terminals as logical unit (LU) names. These terminal (LU) names are shown as labels on the APPL statements and as values for the ACBNAME= parameter.
   - The *application major node table* for PVMG application, which must be activated by VTAM startup. This mode table is shown as a value for the MODETAB=parameter.

3. Tailor the APPL statements in this VTAMLST file as follows:

   a. For the first APPL statement, the value on the label and ACBNAME= parameter must match the LOCAL record definition in the PVMG configuration file.

      For example,

      ACBNAME=N2ECAPV2

   b. The value on the label and the ACBNAME= parameter for each subsequent APPL statements must be within an LU range that you specify on the LU definitions in the PVMG configuration file.

      The ACB definitions start with N2ECAP01 and are in numeric sequence up to the highest number of PVM sessions *through SNA* you wish to support. As distributed, the PVMG VTAMLST file has ACB name

definitions that run from N2ECAP01 through N2ECAP20, allowing a maximum of 20 SNA sessions.

c. The MODETAB= parameter points to the mode table file name which is a table of session parameters. The SNAMODET ASSEMBLE file is distributed as a sample mode table on the PVM product tape. The file contains definitions for your terminal types.

   If you do not wish to use the supplied samples, edit the SNAMODET ASSEMBLE file and assemble it. Next append this file to an existing file that has been globally defined (GLOBAL command) on VTAM's search order or build your own one-member TXTLIB and GLOBAL that into VTAM's search order.

Step 9. Apply Service

Check to see if any programming service should be applied to your system. If service is necessary, apply the service before trying to start your PVM system. See Chapter 11, "Applying Service to PVM 2.1" on page 141 for information on how to apply service.

Step 10. Verify Initial Installation

After you have loaded the PVM distribution tape, perform the following steps to verify the success of PVM installation. You can verify the installation of PVM before or after PVM is tailored to your installation's needs.

To verify the installation of PVM:

1. Log on MAINT user ID.
2. Enter

   ```
   cp detach 36E
   ```

   This guarantees that PVM can link to this disk as its 191 in R/W mode.
3. Log on your PVM user ID.
4. After a short time, you should see this (or a similar display) at your display station:

```
Execution begins...
VM Pass-Through Facility 2.1.0000, built MM/DD/YY HH:MM:SS, is ready
:
```

5. Enter

   ```
   disc
   ```

   This disconnects the PVM user ID.
6. Log on any other valid CMS user ID.
7. After receiving the system ready message, Enter

   ```
   passthru
   ```
8. A PVM Selection Menu should be displayed containing node IDs that were defined in the PVM configuration file.

To verify the installation of MPVM:

1. Leave the PVM Selection Menu
2. Enter

   ```
   mpvm
   ```

   at the Ready prompt.
3. A MPVM Selection Menu should be displayed containing node IDs that were defined in the MPVMSYS file.

Verification is now complete. Press PF3 or PF12 to exit from the MPVM Selection Menu.

To verify installation of PVMG:

1. Enter

   ```
   CP DETACH 38E
   ```

   from the MAINT user ID. This ensures that PVMG can link to this disk in R/W mode.
2. Log on to user ID PVMG. This will IPL GCS, which starts the PROFILE GCS.
3. After a short time, you should see this (or a similar display) at your display station:

```
runpvmg
370 GCS1.1 BUILT 09/10/91
Generated at 02/21/91 15:58:57
GCTACC423I A (0191) R/W
Ready;
09/20/91 18:52:01 EFGMAI040I VM Pass-Through Gateway Services 2.1.0000,
built 09/20/91 16:40:19, is ready
09/20/91 18:52:01 EFGOPE625I The PVMG command processor is ready
```

4. Enter:

   ```
   PVMG DISC
   ```

   This disconnects the PVMG user ID.
5. You should be at a screen to which you can log on. Enter

   ```
   dial VTAM
   ```

   Where VTAM is the virtual machine identifier. Then Enter

   ```
   logon applid(pvmgapplid)
   ```

   where *pvmgapplid* is the VTAM applid of PVMG.

   You should see a PVMG Selection Menu.

Verification of PVMG installation is now complete.

Chapter 5. CVIEW Installation Planning

This chapter describes the distribution media and the operating environment required for the installation of CVIEW, and the requirements for migrating to CVIEW.

CVIEW Virtual Machine Requirements

You must create a directory entry for the CVIEW virtual machine before installing CVIEW. CVIEW requires specific options on several directory definition control statements. The following topics describe the directory-entry requirements.

Disk Space

The minimum disk space requirement for the CVIEW virtual machine A-disk is five (5) cylinders of 3380 disk space, or the equivalent. If you format the 191 A-disk prior to installation, use a block size of 4096 for optimum performance.

Virtual Storage

The CVIEW virtual machine requires 5 megabytes of virtual storage if it is to support up to 30 concurrent users at separate display stations. The number of concurrent users to be supported by the CVIEW virtual machine is selected during customization. See Chapter 17, "Customizing CVIEW,"

Privilege Class

CVIEW requires a privilege class of 'B' or 'G'. Privilege class B provides message no header (MSGNOH). MSGNOH returns messages without the standard header associated with the MESSAGE command to users for CVIEW privileged and general commands. An important consideration to note in assigning privilege class B is that any user specified during customization as a CVIEW privileged user can enter privilege class B CP commands to the CVIEW virtual machine.

Channel Mode and Nonshared Virtual Control Units

You must specify block multiplexer (BMX) mode of channel operation and nonshared virtual control units (VCUNOSHR) on the OPTION control statement. In BMX mode, the virtual channel is not busy until the initial start I/O command (SIO) is complete; this allows the successful start of multiple SIOs to different devices on the same channel. The VCUNOSHR option ensures that the CVIEW virtual machine does not use shared subchannel protocols.

Note: The BMX and VCUNOSHR options are only good under the 370 option.

User ID

For further installation details, see the PVM 2.1.1 Program Directory.

© Copyright IBM Corp. 1991,1993

CVIEW Directory Entry

The control statements in Figure 9 satisfy the directory entry requirements for the CVIEW virtual machine.

```
USER CVIEW XXXXXXXX 5M 5M G  (Note 1)
OPTION BMX VCUNOSHR (Note 2)
ACCOUNT number
IPL CMS
CONSOLE 01F 3215
SPOOL  00C 3505
SPOOL  00D 3525
SPOOL  00E 1403
LINK MAINT 190 190 RR
LINK MAINT 19E 19E RR
LINK P684100E 405 191 MR
```

Figure 9. Control Statements for a CVIEW Directory Entry

Notes:

1. The USER control statement defines the user-ID as CVIEW, the virtual storage as 5 megabytes, and the privilege class as G.
2. The OPTION control statement defines the mode of channel operation as BMX. BMX and VCUNOSHR are not supported in VM/ESA.

Virtual addresses 501 through 530 are reserved for use by CVIEW and cannot be assigned to any resource associated with the CVIEW virtual machine.

For additional information about directory entries, see the *VM/ESA: Planning and Administration, SC24-5521*.

CVIEW Operating Environment

The following section describes what is required to operate CVIEW.

Machine Requirements

CVIEW supports only display stations that are compatible with at least one of the following IBM device types:

- IBM 3277 Model 2 (screen size: 24 X 80)
- IBM 3278 Model 2 (screen size: 24 X 80)
- IBM 3278 Model 3 (screen size: 32 X 80)
- IBM 3278 Model 4 (screen size: 43 X 80)
- IBM 3278 Model 5 (screen size: 27 X 132)

The format of the data used in a CVIEW session must conform to 3270 architecture for local display stations. Extended data streams (EDS), which includes extended color, extended highlighting, programmed symbols, and partitions, are not supported.

CVIEW supports the following IBM display stations in any combination:

- IBM 3277 - Model 2

- IBM 3278 - Models 2, 3, 4, 5
- IBM 3279 - All models when operated in 4-color compatibility mode
- IBM 3178
- IBM 3179
- IBM 3180
- IBM 3290
- IBM 3270 PC - When operated in "host session" mode
- IBM 3270 PC/G - When operated in "host session" mode
- IBM 3270 PC/GX - When operated in "host session" mode
- IBM PC XT/370 - When operated in "remote session" mode

CVIEW requires at least two (2) display stations to establish a session.

Multiple CVIEW Virtual Machines

You can create additional CVIEW virtual machines to support additional networking sessions and to support more than 30 concurrent users. CP allows multiple virtual machines to simultaneously create logical device. If the Pass-Through Facility is installed on your system, it uses a virtual machine that creates logical devices. This chapter presents the steps necessary to create additional CVIEW virtual machines.

Figure 10 shows how additional CVIEW virtual machines relate to the base CVIEW virtual machine.

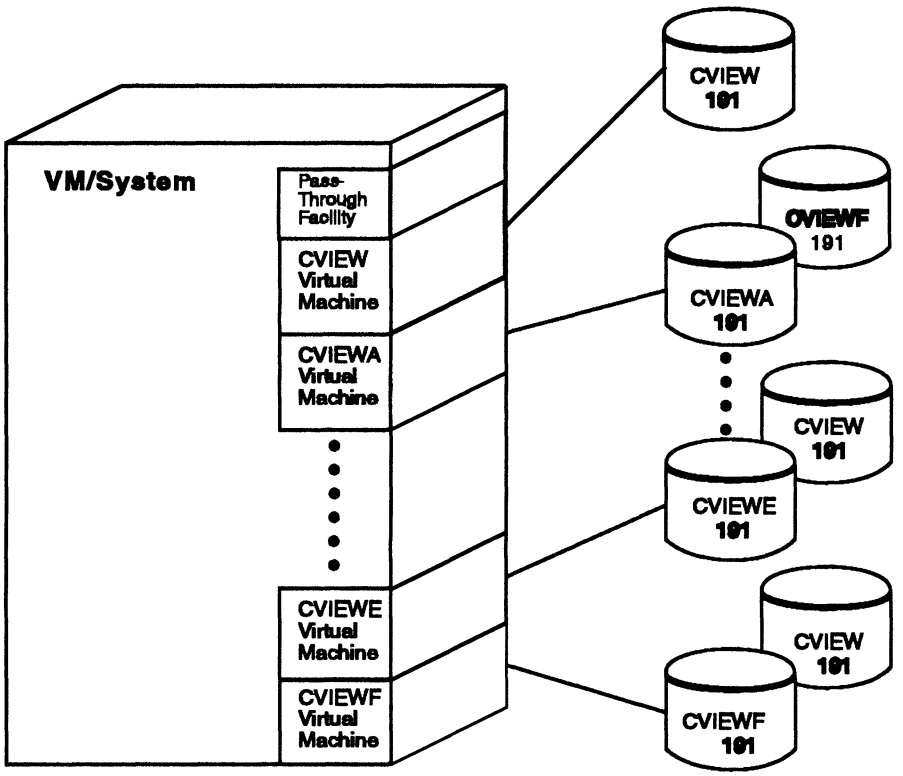


Figure 10. A System with Multiple CVIEW Virtual Machines

To create additional CVIEW virtual machines, follow these steps:

1. You must install CVIEW before you can create additional CVIEW virtual machines. If you have not installed CVIEW, do so now. See the PVM 2.1.1 Program Directory for instructions.
2. You must create a directory entry for each additional CVIEW virtual machine. Each additional virtual machine requires the same directory entries as the base CVIEW virtual machine except for disk space and the user-ID. The 191 A-disk for each additional virtual machine requires only five (5) cylinders. The user-ID cannot be CVIEW. See the PVM 2.1.1 Program Directory for directory entry requirements.
3. Log on the base CVIEW virtual machine. (Enter: L CVIEW)
4. If CVIEW is running, enter: SHUTDOWN
5. Rename the CONSULT IDNAME file on the base CVIEW virtual machine A-disk to change the file mode from A0 to A1. This step allows the customization exec, DXGINIT, to copy the CONSULT IDNAME file to the 191 A-disk for the additional CVIEW virtual machine.
6. Log off the base CVIEW virtual machine.
7. Log on to the user-ID assigned to one of the additional CVIEW virtual machines.
8. Enter: CP LINK CVIEW 191 xxx RR password
9. Enter: AC xxx B/A

Steps 8 and 9 link and access the base CVIEW 191 disk as a read-only extension of the A-disk.

10. The additional CVIEW virtual machine must be customized. Copy the base CONSULT IDNAME file to your new A disk as filemode 0. Enter the DXGINIT command. If you want different listed consultants for the additional virtual machine, respond *No* to the following customization prompt and proceed to step 11.

    Have you entered the consultant IDs in the CONSULT IDNAME file? Enter Yes or No. Press ENTER for YES.

    If your response was *Yes*, proceed to step 12.

11. If your response was *No* for step 10, customization ends. Edit the CONSULT IDNAME file at this time. When you finish editing the CONSULT IDNAME file, enter: DXGINIT and respond Yes.

12. Customization prompts you for the privileged users for the additional virtual machine. If the prompt lists CVIEW, change CVIEW to the user-ID assigned to this additional CVIEW virtual machine.

13. After DXGINIT finishes, edit the PROFILE EXEC to include the following commands:

    ```
    CP LINK CVIEW 191 xxx RR password
    ACC xxx B/A
    EXEC DXGCVIEW
    ```

14. To define another CVIEW virtual machine, log off and return to step 7.

15. Log on to the base CVIEW virtual machine and rename the CONSULT IDNAME file to change the file mode from A1 back to A0.

16. After each additional CVIEW virtual machine has been defined and customized, you can log on each additional virtual machine and start CVIEW. Users can access an additional CVIEW virtual machine by issuing a 'DIAL cviewid' command, where cviewid is the user-ID assigned to the additional CVIEW virtual machine.

Chapter 6. Defining Your PVM Configuration

Before starting PVM, you, as the system administrator, should plan your network, create a configuration file to define your PVM network, and create a PVM profile exec and a PROFILE EXEC to automate system initialization. You will need to do the same tasks for PVMG if you are installing PVMG. If you install MPVM, you must create a systems names file called MPVMSYS NAMES. See Chapter 8, "Developing NAMES Files for MPVM" on page 95 for instructions on creating the MPVMSYS NAMES file.

Also, there are some requirements and conditions you will want to consider before planning your network. See Chapter 3, "Administrative Considerations" on page 21 for this information.

Planning a Network

Before defining your PVM network, it is a good idea to plan out what your network will look like. To help you plan and keep track of your PVM network, you should know the following about your intended network:

- Local PVM system name
- Target system names
- PVMG server names
- Types of connection paths for the target systems
- Port definitions
- Routing paths.

Table 7 on page 65, Table 8 on page 66, and Figure 11 on page 64, illustrate a sample PVM configuration that is defined in this chapter. You may want to create similar tables and diagrams to help you configure your PVM network.

Note: In order to tailor PVM files, you should be logged onto PVM and have R/W access to PVM's 191 minidisk (PVM 2.1 links to MAINT's 36E minidisk as 191 in MR mode, and PVM 2.1.1 links to P684100E's 401 minidisk as 191 in MR mode, through the CP directory). You may have to detach the 36E (or 402) minidisk from MAINT (or P684100E) to gain write access, because M/R mode only allows one R/W link at a time. Once logged on you should shutdown PVM (if running) by issuing:

```
SHUTDOWN IMMED
```

You can then tailor the following files:

- PVM CONFIG
- PROFILE PVM
- MPVMSYS NAMES
- MPVM NAMES (as a sample for users).

© Copyright IBM Corp. 1991,1993

The following table describes the sample files supplied on the PVM 2.1.1 LOCALSAM (2C2) disk for the PVM user id:

Table 6. PVM 2.1.1 Supplied Sample Files for the PVM user id

File Name	Sample Description
PROFILE PVMSAMP	Sample profile PVM file.
PROFPVM EXECSAMP	Sample profile exec file.
PVMSAMP DIRECT	Sample directory file.
PVM CONFSAMP	Sample PVM CONFIG file.
UCOMDIR NAMESAMP	Sample CMS communications directory.

The example PVM network illustration and corresponding planning tables that follow will help you as you plan your PVM network.

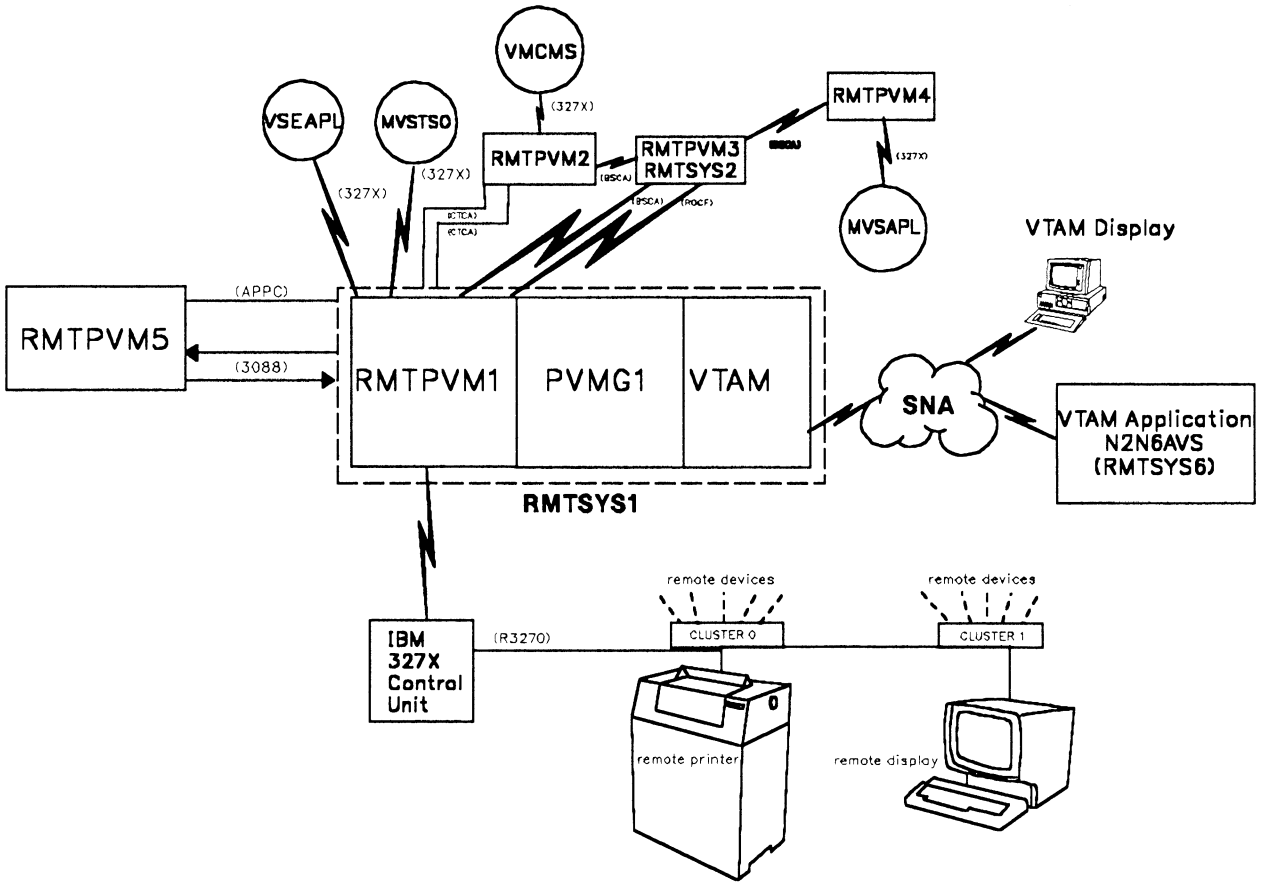


Figure 11. Sample PVM network. Shows the hardware that the PVM profile exec and the PVM configuration files must map.

Table 7. Configuration File Planning Table Number 1. Defines PVM links — systems connected to the local PVM.

Target Name	Connection Type and Address	Comments
VSEAPL	327X, 080	VSEAPL is a virtual storage extended (VSE) system. Define ports on this link for: • An IBM 3278 Model 2 Display Station for exclusive use by user ID USERA on RMTPVM1 • An IBM 3278 Model 4 Display Station at real address 0351 on RMTPVM1 • An IBM 3277 Model 2 Display Station • A 3270 Information Display Station printer at real address 0360 on RMTPVM1.
MVSTSO	327X, 081	MVSTSO is a Multiple Virtual Storage (MVS) system. Define ports on this link for: • An IBM 3278 Model 4 Display Station • A 3270 Information Display Station printer at real address 0360 on RMTPVM1 • A 3270 Information Display Station printer at real address 0310 on RMTPVM2 • A 3270 Information Display Station printer.
PVMG1	IUCV	This links PVMG1 to RMTPVM1
RMTPVM2A	CTCA, 200	This link is part of a line group to RMTPVM2.
RMTPVM2B	CTCA, 201	This link is part of a line group to RMTPVM2.
RMTPVM2	GROUP	Define RMTPVM2A and RMTPVM2B as a line group to RMTPVM2.
RMTPVM5A	3088, 500	This link is part of a line group to RMTPVM5.
RMTPVM5B	APPC	This link is part of a line group to RMTPVM5.
RMTPVM5	GROUP	Define RMTPVM5A and RMTPVM5B as a line group to RMTPVM5.
RMTPVM3	BSCA, 082	
RMTSYS2	ROCF, 2F	Physically, this is the same node as RMTPVM3.
REM3270	R3270, 0B0	This link defines a remote cluster. Cluster 0 has eight ports defined for: • Two IBM 3278 Model 2 Display Stations • Two IBM 3278 Model 3 Display Stations • Two IBM 3278 Model 4 Display Stations • One IBM 3289 printer • One IBM 3286 printer. Cluster 1 has eight ports defined for: • One IBM 3278 Model 2 Display Station • One IBM 3278 Model 3 Display Station • Three IBM 3279 Model 3 Display Stations • One IBM 3284 printer • One IBM 3287 printer • One IBM 3289 printer.

Table 8. Configuration File Planning Table Number 2. Defines PVM routes to other systems in the network and to SNA applications.

Target Name	Route Session Requests Through	Comments
VMCMS	RMTPVM2	
RMTPVM3	RMTPVM2	Even though RMTPVM3 is already linked to RMTPVM1, define an alternate route to it through RMTPVM2. If the links between RMTPVM1 and RMTPVM3 go down, PVM sessions to RMTPVM3 can be started through the alternate route.
RMTPVM2	RMTPVM3	Even though RMTPVM2 is already linked to RMTPVM1, define an alternate route to it through RMTPVM3. If the links between RMTPVM1 and RMTPVM2 go down, PVM sessions to RMTPVM2 can be started through the alternate route.
RMTPVM4	RMTPVM3	Define a route to RMTPVM4 through RMTPVM3.
VMCMS	RMTPVM3	Define a route through RMTPVM3 to RMTPVM2 then to VMCMS.
MVSAPL	RMTPVM3	Define a route through RMTPVM3 to RMTPVM4 then to MVSAPL.
RMTSYS6	PVMG1	This is the PVMG server for this network. **Note**: The VTAM *applid* N2N6AVSC has a node name of RMTSYS6 in the PVM network. The node name could also be known as the VTAM *applid* , in this case, N2N6AVSC. Node names are resolved in the PVMG configuration file.

Creating the Configuration File

You use a PVM configuration file on the PVM user ID to define your network. It contains lines called records. The records tell PVM how the nodes in your network are connected.

There are a few restrictions regarding the configuration file:

- All data in the file must be in uppercase, except for the session termination string specified on the SESSTERM record.
- Input for the file's records is restricted to columns 1 through 71, with no continuation.
- Some records must appear in a particular order See Figure 17 on page 70 for more information on the required order.
- The file type must be CONFIG.

Sample Configuration Files

The following is a configuration file for the PVM system, whose node ID is RMTPVM1, shown in Figure 11 on page 64.

```
*************************************************************
*  Configuration file for node RMTPVM1 - RMTPVM1 CONFIG  *
*************************************************************
LANG AMENG
XDBC CONS
LOCAL RMTPVM1
RESID PVM1
* Set up an emulator link to VSEAPL
LINK 080 VSEAPL 327X
* Ports defined on the emulator link to VSEAPL
PORT 1 3278-2 RMTPVM1 USERA
PORT 2 3278-4 RMTPVM1 0351
PORT 8 3277 *
PORT 9 328X RMTPVM1 0360
* Set up an emulator link to MVSTSO
LINK 081 MVSTSO 327X
* Ports defined on the emulator link to MVSTSO
PORT 0 3278-4 *
PORT 1 328X RMTPVM1 0360
PORT 2 328X RMTPVM2 0310
PORT 3 328X *
* Set up PVM to PVM links
LINK 200 RMTPVM2A CTCA
LINK 201 RMTPVM2B CTCA
LINK 500 RMTPVM5A 3088
LINK EAGLE RMTPVM5B APPC USER SMITH
LINK * PVMG1 IUCV USER PVMG
LINK 082 RMTPVM3 BSCA
* Set up an ROCF link to RMTSYS2 - physically, this is the same
* node as RMTPVM3
LINK 2F RMTSYS2 ROCF
* Set up a link for a R3270 cluster and call it REM3270
LINK 0B0 REM3270 R3270
* Specify that the R3270 control unit at address 0 has
* a maximum of 8 ports
CLUSTER 0 8 3274E
* Define the ports available to REM3270
CLPORT 00 3278-2
CLPORT 01 3278-2
CLPORT 02 3278-3
CLPORT 03 3278-3
CLPORT 04 3278-4
CLPORT 05 3278-4
* Define the seventh port available to RMTPVM2 on line 0B0 to be a
* 3289 printer port.  RMTPVM2 LF00 specifies that the printer will
* be connected to node ID RMTPVM2 at address F00.  The logical
* printer will be attached to virtual machine ID RSCS at virtual
* address 0C0.  The device on this port is actually a 4234 printer.
CLPORT 06 3289 RMTPVM2 LF00 RSCS 0C0
* Define the eighth port available to REM3270 on line 0B0 to be a
* 3286 printer port.  VMCMS 04 specifies that the printer will be
* connected to the system VMCMS at address 04.
CLPORT 07 3286 VMCMS 04
* Specify that the R3270 control unit at address 1 has
* a maximum of 8 ports.
CLUSTER 1 8 3274E
* Define the ports on the cluster defined above.
CLPORT 00 3279-3
CLPORT 01 3279-3
CLPORT 02 3279-3
CLPORT 03 3278-2
CLPORT 04 3278-3
CLPORT 05 3284 MVSTO 3
CLPORT 06 3287 RMTPVM1 LF00 RSCS 0C0
CLPORT 07 3289 RMTPVM1 LF01 RSCS 0C1
GROUP RMTPVM2 RMTPVM2A RMTPVM2B
GROUP RMTPVM5 RMTPVM5A RMTPVM5B
ROUTE RMTSYS6 PVMG1
ROUTE VMCMS RMTPVM2
```

Figure 12 (Part 1 of 2). Sample Configuration File for RMTPVM1 -- RMTPVM1 CONFIG

```
ROUTE RMTPVM3 RMTPVM2
ROUTE DEFAULT RMTPVM3
* The configuration file records to this point are required to
* define all routes from RMTPVM1 to the other nodes.
* The following records might apply to any PVM node.
AUTHORIZ OPR
AUTHORIZ JONES RMTPVM2
SUBSET LISTFILE
AUDIT DATA RECORD
LNODE * AUTO
* Set the PF keys on the Selection Menu.
SELECT RMTPVM1
SELECT VSEAPL
SELECT MVSTSO
SELECT RMTPVM3
SELECT
SELECT RMTPVM5
* Set time restrictions.
STATTIME 1200
MAXIDLE 6000
MAXSESS 4
MAXSUSP 1500
MAXUSER 600
MAXTDISC 3600
TDISC 600
TIMEOUT 500
MSGNOH
* Specify that a VMDUMP is to be taken if abnormal termination of
* PVM occurs.  The VMDUMP option should be specified if either the
* PVM/IPCS or the VM/ESA or VM/XA Dump Viewing Facility is to be used.
DUMP VMDUMP OPERATOR
* The BUFFERS, EXT, and I/O entries are not illustrated in this
* sample configuration
MSGOPTNS Y Y Y Y
```

Figure 12 (Part 2 of 2). Sample Configuration File for RMTPVM1 -- RMTPVM1 CONFIG

The following are partial samples of PVM configuration files for the other PVM nodes in the sample network. The records following the ROUTE records are not shown in these samples because they could be similar to the ones in the configuration file for RMTPVM1.

```
************************************************************
*  Configuration file for node RMTPVM2 - RMTPVM2 CONFIG  *
************************************************************
LOCAL RMTPVM2
LINK 0A0 RMTPVM3 BSCA
LINK 0A1 VMCMS 327X
PORT 0 3278-2 RMTPVM2 USERX
PORT 1 3278-2 *
PORT 2 328X RMTPVM2 0380
PORT 3 328X RMTPVM3 0370
PORT 4 3287 *
LINK 300 RMTPVM1A CTCA
LINK 301 RMTPVM1B CTCA
GROUP RMTPVM1 RMTPVM1A RMTPVM1B
ROUTE VSEAPL RMTPVM1
ROUTE MVSTSO RMTPVM1
ROUTE RMTPVM5 RMTPVM1
ROUTE RMTPVM3 RMTPVM1
ROUTE RMTPVM1 RMTPVM3
ROUTE RMTPVM4 RMTPVM3
ROUTE MVSAPL RMTPVM3
ROUTE RMTSYS6 RMTPVM1
```

Figure 13. Sample Configuration File for a Target System -- RMTPVM2 CONFIG

```
************************************************************
*  Configuration file for node RMTPVM3 - RMTPVM3 CONFIG  *
************************************************************
LOCAL RMTPVM3
LINK 090 RMTPVM1 BSCA
LINK 091 RMTPVM2 BSCA
LINK 092 RMTPVM4 BSCA
ROUTE VSEAPL RMTPVM1
ROUTE MVSTSO RMTPVM1
ROUTE RMTPVM2 RMTPVM1
ROUTE RMTPVM5 RMTPVM1
ROUTE RMTPVM1 RMTPVM2
ROUTE VMCMS RMTPVM2
ROUTE MVSAPL RMTPVM4
ROUTE RMTSYS6 RMTPVM1
```

Figure 14. Sample Configuration File for an Intermediate Node -- RMTPVM3 CONFIG

```
************************************************************
*  Configuration file for node RMTPVM4 - RMTPVM4 CONFIG  *
************************************************************
LOCAL RMTPVM4
LINK 080 RMTPVM3 BSCA
LINK 081 MVSAPL 327X
PORT 0 3278-4 RMTPVM4 USERY
PORT 1 3278-4 *
PORT 2 328X RMTPVM3 0370
ROUTE RMTPVM2 RMTPVM3
ROUTE RMTPVM1 RMTPVM3
ROUTE VSEAPL RMTPVM3
ROUTE MVSTSO RMTPVM3
ROUTE RMTPVM5 RMTPVM3
ROUTE VMCMS RMTPVM3
ROUTE RMTSYS6 RMTPVM3
```

Figure 15. Sample Configuration File for a Target Node Routed through an Intermediate Node -- RMTPVM4 CONFIG

```
************************************************************
*  Configuration file for node RMTPVM5 - RMTPVM5 CONFIG  *
************************************************************
LOCAL RMTPVM5
RESID PVM5
LINK 600 RMTPVM1C 3088
LINK HAWK RMTPVM1D APPC USER JONES
GROUP RMTPVM1 RMTPVM1C RMTPVM1D
ROUTE RMTPVM2 RMTPVM1
ROUTE RMTPVM3 RMTPVM1
ROUTE RMTPVM4 RMTPVM1
ROUTE MVSAPL RMTPVM1
ROUTE VSEAPL RMTPVM1
ROUTE MVSTSO RMTPVM1
ROUTE VMCMS RMTPVM1
ROUTE RMTSYS6 RMTPVM1
```

Figure 16. Sample Configuration File for a Target Node Routed through an Intermediate Node -- RMTPVM5 CONFIG

The following sections describe the configuration file records for the sample configuration file for RMTPVM1. You can find the full record format and parameter descriptions in Appendix D, "PVM Configuration File Records." The records are introduced in the order that they appear in the configuration file for the sample PVM system. Figure 17 illustrates the order of records in a PVM configuration file.

```
LOCAL
LINK
PORT
PORT
LINK
CLUSTER
CLPORT
CLUSTER
CLPORT
CLPORT
CLPORT
LINK
GROUP
ROUTE
ROUTE
AUTHORIZ
```

Figure 17. Order of Configuration File Records

The AUDIT, AUTO, BUFFERS, DUMP, EXT, I/O, LNODE, MAXCONN, MAXIDLE, MAXSESS, MAXSUSP, MAXTDISC, MAXUSER, MSGNOH, MSGOPTNS, OPID, PVMPROP, SELECT, SESSTERM, STATTIME, SUBSET, TDSIC, and TIMEOUT records may follow in any order following the last AUTHORIZE record.

The *, LANG, and XDBC records can be placed anywhere in the configuration file.

The RESID record may follow anywhere in the configuration file after the LOCAL record.

Commenting the PVM Configuration File

Use the comment (*) record to put comments in the configuration file. For example,

```
*  Configuration file for node RMTPVM1 - RMTPVM1 CONFIG
```

is the first record of the configuration file for RMTPVM1. The asterisk (*) must be in column 1.

Comment records can be placed anywhere in the configuration file.

Identifying the System Language

Use the LANG record to identify the default national language for a node. For example,

```
LANG AMENG
```

assigns American English as the default national language for RMTPVM1.

The LANG record can be placed anywhere in the configuration file.

Excluding Double-Byte Character Messages

Use the XDBC record to exclude users from receiving DBCS messages. For example, if the LANG record identifies a double-byte character language such as KANJI, and the PVM console should not receive a DBCS message, enter the following:

```
XDBC CONS
```

Now any DBCS messages received by PVM for the console are intercepted and the equivalent message in the backup language is sent to the console in place of the DBCS message. The AUDIT file also receives DBCS message in the backup language.

See "QUERY" on page 318 for information on the DBCS settings and "CDBC" on page 289 for information on changing DBCS-exclusion settings.

The XDBC record can be placed anywhere in the configuration file.

Identifying the Local Node Name

Use the LOCAL record to identify the local node name. For example,

```
LOCAL RMTPVM1
```

assigns the node ID RMTPVM1 to the local PVM system.

The LOCAL record must precede any other records in the configuration file except for comment records, the LANG record, or the XDBC record.

Identifying a Unique Resource Name

Use the RESID record to define an APPC/VM resource identifier. For example,

```
RESID PVM1
```

assigns PVM1 as the identifier for node RMTPVM1. It is used as the resource identifier in the UCOMDIR NAMES file of other PVM nodes, not the local node. (:tpn. is the tag used in the UCOMDIR NAMES file to identify this resource. It is the target identifier for APPC links.)

The RESID record must follow the LOCAL record.

Defining a Link from a Local Node to Another Node

Use the LINK record to define a link from the local node to another node, or to a PVMG node.

For example,

```
LINK 080 VSEAPL 327X
LINK 081 MVSTSO 327X
LINK 200 RMTPVM2A CTCA
LINK 201 RMTPVM2B CTCA
LINK 500 RMTPVM5A 3088
LINK EAGLE RMTPVM5B APPC USER SMITH
LINK 082 RMTPVM3 BSCA
LINK 2F  RMTSYS2 ROCF
LINK 0B0 REM3270 R3270
LINK *   PVMG1 IUCV USER PVMG
```

are the links defined from node RMTPVM1.

- A 327x connection with a line address of 080 connects RMTPVM1 with VSEAPL. Also, a 327x connection with a line address of 081 connects RMTPVM1 with MVSTSO.
- PVM to PVM connections are made from RMTPVM1 to RMTPVM2A (address 200, type CTCA), from RMTPVM1 to RMTPVM2B (address 201, type CTCA), from RMTPVM1 to RMTPVM5A (address 500, type 3088), and from RMTPVM1 to RMTPVM3 (address 082, type BSCA).
- EAGLE refers to a nickname entry that is found in UCOMDIR NAMES. The entry defines the target node to connect to.
- The RMTPVM1 link to RMTSYS2 allows authorized users to connect to the RMTSYS2 ROCF.
- Link REM3270 defines the R3270 devices connected to a BSC 327x Control Unit.
- An IUCV connection to the PVMG machine, PVMG1, at the VM user ID PVMG.

LINK records must follow the LOCAL record.

Adding Devices to a Line

LINK records defining a connection other than a PVM to PVM connection must be followed by records defining the ports on that connection. For a 327x type link, the ports are defined by PORT records; for an R3270 type link, ports are defined by CLUSTER and CLPORT records.

Adding Devices to a 327x Line

Use the PORT to add devices to a 327x type link.

For example, the sample network's VSEAPL node has these ports defined:

```
PORT 0 3278-2 RMTPVM1 USERA
PORT 1 3278-2 RMTPVM1 0351
PORT 8 3278 *
PORT 9 328X RMTPVM1 0360
```

Port 0 is defined as a 3278 Model 2 Display Station reserved for exclusive use by the user identified as USERA at node RMTPVM1. Port 1 is defined as a 3278 Model 2 Display Station reserved for exclusive use by the user at the local display station whose real address is 351. Port 8 is defined as a 3278 Display Station accessible to any user. Port 9 is defined as a 3270 Information Display System printer reserved for exclusive use by a printer whose real address is 360.

PORT records for the node MVSTSO are shown in the complete sample configuration file shown on page 66.

PORT records must follow the LINK record with which they are associated.

Adding Devices to an R3270 Line

Use the CLUSTER and CLPORT records to add devices to a R3270 type link. The CLUSTER record defines a remote 3271 or a remote 3274 control unit on a line driven by the PVM R3270 line driver.

For example, CLUSTER records defined for the node REM3270 are:

```
CLUSTER 0 8 3274E
CLUSTER 1 8 3274E
```

Both CLUSTER records indicate that there are 8 ports for each control unit and that the control units support 3270 extended data streams.

CLUSTER records must follow the LINK record for the line they are defining.

The CLPORT record defines the ports on the control unit associated with a CLUSTER record. For example, CLPORT records associated with the CLUSTER 0 record are:

```
CLPORT 00 3278-2
CLPORT 01 3278-2
CLPORT 02 3278-3
CLPORT 03 3278-3
CLPORT 04 3278-4
CLPORT 05 3278-4
CLPORT 06 3289 RMTPVM2 LF00 RSCS 0C0
CLPORT 07 3286 VMCMS 04
```

CLPORT records 00 through 05 define various 3278 Display Station models to the first 6 ports available to REM3270 on line 0B0.

CLPORT 06 defines the seventh port available to REM3270 on line 0B0 to be a 3289 printer port. The parameters RMTPVM2 and F00 specify that the printer will be connected to node ID RMTPVM2 at address logical F00. The logical printer will be attached to virtual machine ID RSCS at virtual address 0C0.

CLPORT 07 defines the eighth port available to REM3270 on line 0B0 to be a 3286 printer port. The parameters VMCMS and 04 specify that the printer will be connected to the system VMCMS at port address 04.

CLPORT records must follow the associated CLUSTER record that defines the control unit the device is attached to.

Defining Groups

Use the GROUP record to form a group from already defined links. For example,

```
GROUP RMTPVM2 RMTPVM2A RMTPVM2B
GROUP RMTPVM5 RMTPVM5A RMTPVM5B
```

forms a group named RMTPVM2 containing CTCA link RMTPVM2A and CTCA link RMTPVM2B. Group RMTPVM5 contains the APPC link RMTPVM5B and 3088 link RMTPVM5A.

You can also use the following GROUP records to form the same groups as defined above:

```
GROUP RMTPVM2 RMTPVM2A
GROUP RMTPVM2 RMTPVM2B
GROUP RMTPVM5 RMTPVM5A
GROUP RMTPVM5 RMTPVM5B
```

GROUP records must follow all the LOCAL, LINK, CLUSTER, PORT, and CLPORT records.

Defining Routes and Alternate Routes

Use the ROUTE record to define a routing path from the local PVM node to a specified target node. For example, ROUTE records for RMTPVM1 are:

```
ROUTE RMTSYS6 PVMG1
ROUTE VMCMS RMTPVM2
ROUTE RMTPVM3 RMTPVM2
ROUTE DEFAULT RMTPVM3
```

The ROUTE records define routing paths from RMTPVM1 to:

- N2N6AVSC VTAM application routed through PVMG1 to the local PVMG node
- VMCMS through RMTPVM2.
- RMTPVM3 through RMTPVM2. This is an alternate path defined in case the usual BSCA route to RMTPVM3 is not available. The direct route to RMTPVM3 is defined by the following LINK statement (this statement is described above):

  ```
  LINK 082 RMTPVM3 BSCA
  ```

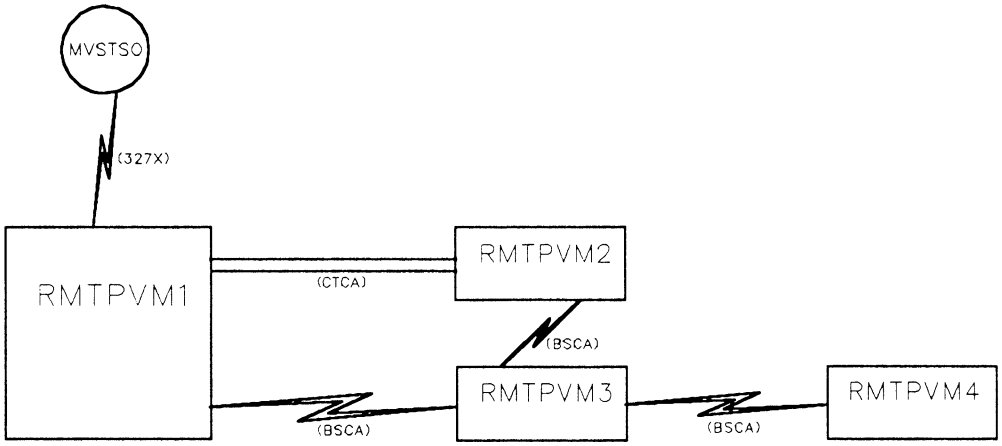
- RMTPVM2, RMTPVM4, and MVSAPL through RMTPVM3 because of the DEFAULT keyword. The path to RMTPVM2 is an alternate path defined in case the usual CTCA route to RMTPVM2 is not available. RMTPVM3's configuration file must define the remaining linkage to MVSAPL through RMTPVM4.

ROUTE records must immediately follow all LINK, PORT, CLUSTER, CLPORT, and GROUP records.

PVM uses the LINK and ROUTE records to determine the path used to a target node. The procedure is:

1. If the target is a node defined by a LINK or GROUP record, and the link is currently connected, that path is used.
2. If the target node is not defined on a LINK, GROUP, or ROUTE record and a default route is specified on the ROUTE record using the DEFAULT keyword, then the session is routed through the specified default node.
3. The routing list is searched for an entry that has the target system node as an end link. If an entry is not found, the session cannot be completed. If the entry is found, the node ID of the *intermediate node* is checked to see if the direct link to the desired target node is connected.
   a. If so, that path is used.
   b. If not, the routing list is searched for a ROUTE entry (alternate route) for the *intermediate node*. If found, the route is checked to see if this link is connected. If connected, the session is routed through this alternate link. If the route entry is not found, session initiation cannot be completed.

For example, in the network shown below, if a user at node RMTPVM3 attempts to access MVSTSO, the routing table is searched because there is no direct link to MVSTSO. The routing table indicates that the path is through node RMTPVM1. If the BSCA link to RMTPVM1 is not connected, the routing table is searched for an entry for RMTPVM1. The table indicates that the alternate route to RMTPVM1 is through node RMTPVM2. If this link is connected, the session is directed through RMTPVM2.



If, at a later time the BSCA link between RMTPVM1 and RMTPVM3 becomes available, all new sessions pass through it. Sessions currently using the alternate route are not effected.

This type of routing causes changes in the routing tables for subsequent PVM nodes.

To illustrate, take another example, with the BSCA link to RMTPVM1 down. A user at RMTPVM4 attempts to access MVSTSO. The session request is routed first to RMTPVM3, and then to RMTPVM2 (because the link from RMTPVM3 to RMTPVM1 is down). RMTPVM2 forward the request to RMTPVM1.

Alternate paths are specified in the PVM configuration file. Care must be taken when specifying ROUTE records. The system programmers must ensure that the routing is consistent among the PVM nodes, and that no routing loops occur.

Authorizing the PVM Operator

Use the AUTHORIZ record to identify users allowed to enter the restricted subset of PVM commands and the nodes from which these commands may be issued. For example,

```
AUTHORIZ OPR
AUTHORIZ JONES RMTPVM2
```

authorizes the person identified by the user ID OPR to enter restricted PVM commands from the local node, RMTPVM1. The second AUTHORIZ statement authorizes user JONES to enter restricted PVM commands from node RMTPVM2 to RMTPVM1.

AUTHORIZ records must follow all LOCAL, LINK, PORT, CLUSTER, CLPORT, GROUP, and ROUTE records.

Identifying CMS Subset Mode Commands to PVM

Use the SUBSET record to identify to PVM command names that will be processed as CMS Subset mode commands. For example,

```
SUBSET LISTFILE
```

informs PVM that the CMS LISTFILE command will be processed as a CMS Subset mode command. See "SUBSET" on page 372 for more information on the SUBSET record.

Note: Using the SUBSET record may degrade PVM performance.

SUBSET records must follow all LOCAL, LINK, PORT, CLUSTER, CLPORT, GROUP, ROUTE, and AUTHORIZ records. All SUBSET records must appear together.

Automatically Connecting Users to a Node

Use of the LNODE serves two purposes when used with:

- Only one operand (a *nodeid*), the *nodeid* is stored by PVM in the destination field of the Initial Menu. This makes it easier for the users receiving an Initial Menu to select that node.
- Two operands (a node ID and the AUTO keyword), then all instances of when either the Initial Menu or the Selection Screen would be presented to the user are changed so that the user is given automatic access to the node specified by the nodeid operand. In this case, the installed system will find that the only instance of when a user is not given automatic access to the specified nodeid is when a CMS user starts the PASSTHRU command specifying a specific nodeid that is different from the nodeid specified in the LNODE record. For example:

  ```
  LNODE * AUTO
  ```

  automatically connects all users to RMTPVM1 (assumed to be the local node name). PVM will bypass presenting both the Initial Menu and Selection Menu. The system logo screen from RMTPVM1 is displayed.

  LNODE records must follow all LOCAL, LINK, PORT, CLUSTER, CLPORT, GROUP, ROUTE and AUTHORIZ records.

Setting PF Keys for PVM Menus

Use the SELECT record to assign node names to PF keys on the PVM Selection Menu and the Initial Menu.

For example, the records:

```
SELECT RMTPVM1
SELECT VSEAPL
SELECT MVSTSO
SELECT RMTPVM3
SELECT
SELECT RMTPVM5
```

assign

- RMTPVM1 to PF1
- VSEAPL to PF2
- MVSTSO to PF3
- RMTPVM3 to PF4
- nothing to PF5 (PF5 is skipped)
- RMTPVM5 to PF6.

SELECT records must follow all LOCAL, LINK, PORT, CLUSTER, CLPORT, ROUTE, GROUP, and AUTHORIZ records.

Saving PVM Information

The AUDIT and DUMP records save information about PVM.

Use the AUDIT record to save PVM console data in a CMS file. This log can be helpful in finding solutions to problems with your PVM system. For example,

```
AUDIT DATA RECORD
```

causes the console data to be stored in a CMS file named DATA RECORD. The first record in the audit file contains a pointer to the last record written. If the audit file, DATA RECORD is full, the data wraps to the beginning of the file, overlaying the existing records. You should specify a maximum size for the audit file that prevents the PVM minidisk from getting full.

Use the DUMP record to specify the type of dump to be taken if abnormal termination of PVM occurs. For example, the record:

```
DUMP VMDUMP
```

causes a VMDUMP dump to be taken of all PVM storage. This type of dump must be specified if you process dumps using either PVM/IPCS, the VM/ESA or VM/XA Dump Viewing Facility.

All AUDIT or DUMP records must follow all LOCAL, LINK, PORT, CLUSTER, CLPORT, ROUTE, GROUP, and AUTHORIZ records.

Specifying Restrictions

There are eight records that are used to specify various restrictions for certain events in PVM.

Use the STATTIME record to specify the number of seconds between calls to the Link Statistics Exit (DVMNMEL). For example, the following record tells PVM to call the Link Statistics Exit every 1200 seconds.

```
STATTIME 1200
```

Use the TDISC record to specify how long a PASSTHRU user is allowed to remain temporarily disconnected from a remote application. For example, the record:

```
TDISC 600
```

lets users remain temporarily disconnected for 600 seconds before PVM terminates the session.

Use the TIMEOUT record to specify the interval between status checks of remote nodes on emulator and network links. For example, the record:

```
TIMEOUT 500
```

specifies that emulator and network line drivers should check the status of the remote nodes at 500-second intervals.

Note: The value specified on the TIMEOUT record determines if a link is still active. The method of testing varies with line driver type. Each emulator and network line driver sets a timer to the TIMEOUT value when the associated link is first connected.

Time expiration is handled by PVM line drivers as follows:

Emulator line driver — A new time interval is set if the other end of the link has polled the line within the current time interval. Otherwise, the line driver is terminated with an indication of a line time out condition.

Network line driver — A new time interval is set if line activity occurs during the current interval. Otherwise, a new time interval is set, and a transmission to the remote end is attempted. If this transmission is not successful, the line driver is terminated with an indication of a line time out condition.

ROCF line driver — The ROCF line driver does not use the TIMEOUT value specified in the configuration file. It uses a self-defined interval of 1 minute. If no activity has occurred on the line when the time expires, the connection is broken. The ROCF line driver simulates the action of hanging up the telephone receiver.

You may also set restrictions for MPVM users by using the:

- MAXSESS record, which specifies the maximum number of sessions a MPVM user may establish (see "MAXSESS" on page 363)
- MAXIDLE record, which specifies the maximum time a MPVM user may remain idle before PVM terminates all of the user's sessions (see "MAXIDLE" on page 362) A user is considered idle when there is no communication from the MPVM user to PVM for any session.
- MAXSUSP record, which specifies the maximum time a MPVM user may have a session suspended before that session is terminated by PVM (see "MAXSUSP" on page 363)
- MAXTDISC record, which specifies the maximum time a MPVM user may remain temporarily disconnected before all of the user's sessions are terminated by PVM (see "MAXTDISC" on page 364)
- MAXUSERS record, which specifies the maximum number of MPVM users who can connect to PVM (see "MAXUSERS" on page 364).

All STATTIME, MAXSESS, MAXSUSP, MAXIDLE, MAXUSER, MAXTDISC, TDISC, or TIMEOUT records must follow all LOCAL, LINK, PORT, CLUSTER, CLPORT, ROUTE, GROUP, and AUTHORIZ records.

Specifying Message Requirements

You can use the MSGNOH and the MSGOPTNS records to indicate how you want messages displayed. Use the MSGNOH record to indicate that responses to all commands entered by the CP SMSG command must be returned using the CP MSGNOH command rather than the CP MSG command.

Use the MSGOPTNS record to determine whether the time stamps or message numbers for messages will be directed to the PVM user ID, the audit file, the users, and/or the operator. For example, the record:

```
MSGOPTNS Y Y Y Y
```

tells PVM to specify the time stamps and the message numbers on the messages to the PVM user ID and the audit file.

MSGOPTNS records must follow all LOCAL, LINK, PORT, CLUSTER, CLPORT, ROUTE, GROUP, and AUTHORIZ records.

Specifying Storage Requirements

Use the BUFFERS, EXT, and I/O records to get more storage for your PVM system when the default is not enough. For more information on these records and how to use them, see "Storage Considerations" on page 25

All BUFFERS, EXT, and I/O records must follow all LOCAL, LINK, PORT, CLUSTER, CLPORT, ROUTE, GROUP, and AUTHORIZ records.

Creating a CMS PROFILE EXEC for PVM

The PROFILE EXEC file has CP and CMS commands in it. You should include the SET RUN ON command in your PROFILE EXEC so when you disconnect from the PVM virtual machine and log on again, you are immediately put back into the PVM environment and receive the log messages and a reconnected message. You do not have to enter the CP BEGIN command to receive these messages.

To automate the initialization of PVM, the final statement in the CMS PROFILE EXEC for PVM must invoke the RUNPVM command supplied with the product. If you do not use the default names for the PVM module (PVM MODULE), the configuration file (PVM CONFIG), and the PVM profile exec (PROFILE PVM), you must start the RUNPVM command in its correct format. For more information on the format of the RUNPVM command, see "RUNPVM" on page 325.

A sample PROFILE EXEC is provided with the distribution tape that you can update for your installation. Your PROFILE EXEC will be similar to the following (the sample below assumes the default names are used):

```
/* This exec is used for automatic startup of PVM Version 2 */
Address 'COMMAND'
TRACE I
'CP SPOOL CONSOLE TO MAINT START'
'CP SET RUN ON'

/*********************************************************************/
/* Attach the necessary lines to our user ID - this assumes that */
/* we have the necessary class 'B' to perform this task.         */
/*********************************************************************/
'CP ATTACH 080 * 080 '
'CP ATTACH 081 * 081 '
'CP ATTACH 200 * 200 '
'CP ATTACH 300 * 201 '
'CP ATTACH 500 * 500 '
'CP ATTACH 501 * 501 '
'CP ATTACH 082 * 082 '
'CP ATTACH 02F * 02F '
'CP ATTACH 0B0 * 0B0 '

'CP SPOOL CONSOLE STOP CLOSE'

/* Get PVM up and running   */
'EXEC RUNPVM'

exit
```

Figure 18. Sample PROFILE EXEC for node RMTPVM1

Creating a CMS Communications Directory

To set up an APPC link, you must load the CMS communications directory into PVM virtual machine's storage. A CMS communications directory is a special NAMES file. It defines symbolic destination names for target nodes and resource names. This file must have a file type of NAMES, and the file name can be any valid eight-character name. The user default file name is UCOMDIR.

The CMS communications directory is reloaded when you start PVM using the RUNPVM command or when you IPL CMS (Release 6 level of CMS or higher).

The following table describes the entry tags contained in the UCOMDIR NAMES file for PVM virtual machine.

Entry Tag	Meaning
:nick.	defines the symbolic destination name for the target node.
:luname.	specifies the name of the global gateway on this node and the name of the global gateway at the target node. (The system administrator defines the names of the global gateways.)
:tpn.	specifies the resource ID of the target node. This is the same name found in the RESID record of the target configuration file. Or, if the target configuration file does not contain a RESID record, :tpn. is the PVM node ID.
:modename.	specifies the mode name for the SNA session connecting the gateway and the target nodes.
:security.	specifies the security level of the conversation. You must use the security level of PGM or SAME and specify the user ID and password entries.
:userid.	specifies the user ID at the target node.
:password.	specifies the password of the user at the target node.

For security reasons, you may not want to have the user ID and password information in the UCOMDIR NAMES file. To avoid this situation, you can place the user ID and password in the requester virtual machine's CP directory entry using the APPCPASS directory statement.

The following figures are samples of CMS communications directories for node RMTPVM1 and node RMTPVM5.

For information on the APPCPASS directory statement and for details on creating CMS communications directories, see *VM/ESA Connectivity Planning, Administration, and Operation SC24-5448*.

```
:nick.EAGLE    :luname.PVMGATE1 PVMGATE5
               :tpn.PVM5
               :modename.VMINT
               :security.PGM
               :userid.JONES
               :password.IANIAN
```

Figure 19. Sample CMS Communications Directory for RMTPVM1

```
:nick.HAWK      :luname.PVMGATE5 PVMGATE1
                :tpn.PVM1
                :modename.VMINT
                :security.PGM
                :userid.SMITH
                :password.WALTER
```

Figure 20. Sample CMS Communications Directory for RMTPVM5

Creating a PVM Profile Exec

The PVM profile exec has PVM commands in it. It provides a way to automate many of the daily operations of starting lines or devices. A PVM profile exec must have a filetype of PVM.

A sample PVM profile exec that you can modify, called PROFILE PVMSAMP, is provided with the distribution tape. This file is required for fully automated system initialization. It issues START commands for the defined lines, and by issuing START GRAF (attached display support) commands, allows users to access PVM through the CP DIAL command. If it exists, the PVM profile exec is executed during system initialization.

The information in the examples below corresponds to the sample network illustrated in Figure 11 on page 64. The examples illustrate how PROFILE PVM files for the PVM systems in the sample might be written. Line addresses on the START LINE entries must match those in the configuration file LINK records for the associated PVM system. These addresses must be from X'20' to X'FF' for telecommunication lines and from X'100' to X'FFF' for CTCA and 3088 links. Addresses on the START GRAF and START PRT entries must be from X'20' to X'FFF'. To start a PCCF task, use the PCCF option on the START command.

PVM profile execs must have a file type of PVM. The PVM profile exec files for the PVM VM systems illustrated in Figure 11 on page 64 might be:

```
*****************************************************
*  PVM Profile Exec for node RMTPVM1 - RMTPVM1 PVM  *
*****************************************************
START LINE 080
START LINE 081
START LINE 082
START LINE 02F
START LINE 0B0
START LINE 200
START LINE 201
START LINE 500
START NODE RMTPVM5B
START NODE PVMG1
START GRAF 450
START GRAF 451
START GRAF 452
START GRAF 453
START GRAF 454
START PRT  491 VSEAPL
*********************************************************************
*  The port number is not specified here because the configuration
*  file's PORT statement thoroughly defines the port's reserved use.
*  If that were not the case, it is recommended that information be
*  included on the START PRT command.
*  Refer to further comments under "Configuration File PORT Record" on page 31.
*********************************************************************
START PCCF JOEUSER
(etc.)
```

Figure 21. PVM Profile Exec for RMTPVM1 -- RMTPVM1 PVM. Sample for the local system.

```
*****************************************************
*  PVM Profile Exec for node RMTPVM2 - RMTPVM2 PVM  *
*****************************************************
START LINE 0A0
START LINE 0A1
START LINE 300
START LINE 301
START GRAF 460
START GRAF 461
START PRT  4A0 MVSTSO
(etc.)
```

Figure 22. PVM Profile Exec for RMTPVM2 -- RMTPVM2 PVM. Sample for a target system.

```
*****************************************************
*  PVM Profile Exec for node RMTPVM3 - RMTPVM3 PVM  *
*****************************************************
START LINE 090
START LINE 091
START LINE 092
START GRAF 470
START PRT  4B1 MVSAPL
(etc.)
```

Figure 23. PVM Profile Exec for RMTPVM3 -- RMTPVM3 PVM. Sample for an intermediate node.

```
*****************************************************
*  PVM Profile Exec for node RMTPVM4 - RMTPVM4 PVM  *
*****************************************************
START LINE 080
START LINE 081
START GRAF 480
(etc.)
```

Figure 24. PVM Profile Exec for RMTPVM4 -- RMTPVM4 PVM. Sample for a target node routed through an intermediate node.

```
*****************************************************
*  PVM Profile Exec for node RMTPVM5 - RMTPVM5 PVM  *
*****************************************************
START LINE 600
START NODE RMTPVM1D
START GRAF 490
(etc.)
```

Figure 25. PVM Profile Exec for RMTPVM5 -- RMTPVM5 PVM. Sample for a target node routed through an intermediate node.

Chapter 7. Defining Your PVMG Configuration

Before starting PVMG, you, as the system administrator, should plan where the PVMG server fits into the PVM network, and create a PVMG CONFIG configuration file that defines your PVMG configuration. You should also create a PVMG PROFILE EXEC to automate PVMG initialization.

There are other requirements and conditions you will want to consider before planning your PVMG configuration. See Chapter 3, "Administrative Considerations" on page 21 for this information.

Planning Your PVMG Configuration

You use the PVMG configuration file on the PVMG user ID to define your PVMG connections to VTAM and PVM. The PVMG CONFIG file contains lines called records. The records tell PVMG how PVM and SNA applications are connected. The PVMG CONFIG file is processed during PVMG initialization. There are a few restrictions regarding the configuration file:

- It is a fixed length 80 byte file
- Except for comments, and session termination string values on the SESSTERM record, all input must be in uppercase
- Input for the file's records is restricted to a single line, and there is no continuation
- Operands must be separated by one or more blanks
- The file type must be CONFIG.

Note: In order to tailor PVMG files, you should be logged onto PVMG and have R/W access to PVMG's 191 minidisk (PVMG links to MAINT's 38E minidisk as 191 in MR mode for PVM 2.1, and links to P684100E's 402 minidisk as 191 in MR mode for PVM 2.1.1, through the CP directory). You may have to detach the 38E (or 402) minidisk from MAINT (or P684100E) to gain write access, because M/R mode only allows one R/W link at a time. Once logged on you should shutdown PVMG (if running) by issuing:

```
PVMG SHUTDOWN IMMED
```

Note: When entering PVMG commands from the PVMG user ID, you must prefix PVMG commands with `pvmg`.

You can now tailor:

- PVMG CONFIG
- PROFILE GCS.

© Copyright IBM Corp. 1991,1993

The following table describes the sample files supplied on the PVM 2.1.1 LOCALSAM (2C2) disk for the PVMG user id:

Table 9. PVM 2.1.1 Supplied Sample Files for the PVMG user id

File Name	Sample Description
PROFILE GCSSAMP	Sample profile GCS file.
PVMG CONFSAMP	Sample PVMG CONFIG file.
PVMG VTAMLST	Sample VTAMLST file for the PVMG component.
PVMGSAMP DIRECT	Sample directory file.

Planning the PVMG Server's Role in the PVM Network

Table 10. Configuration File Planning Table Number 1. Defines PVMG links — systems connected to the local PVMG.

Target Name	Connection Type and Address	Comments
RMTPVM1	IUCV	This is the link to RMTPVM1

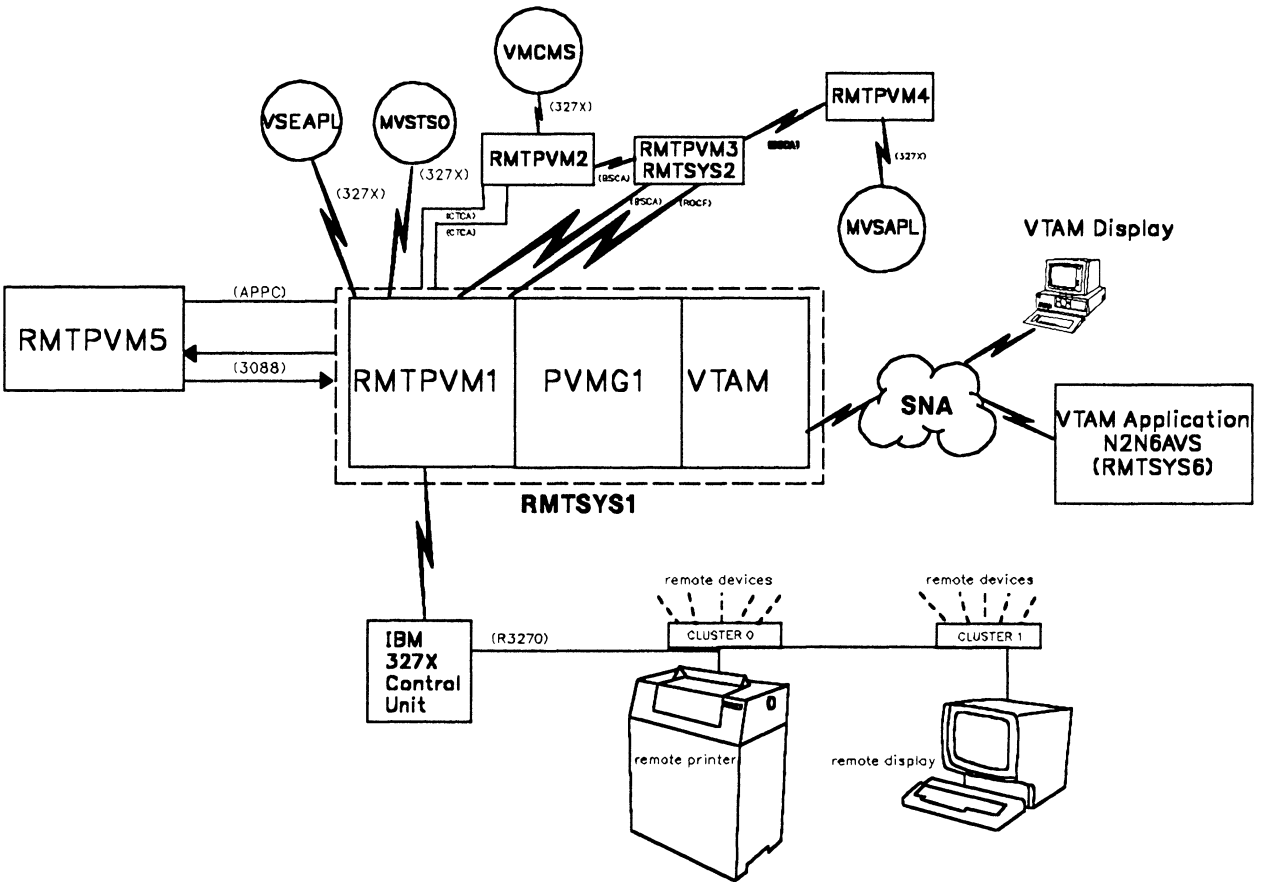


Figure 26. Sample PVM network. Shows the connections that the PVMG profile exec and the PVMG configuration files must map.

Table 11. Configuration File Planning Table Number 2. Defines PVMG routes to other systems in the network.

Target Name	Route Session Requests Through	Comments
RMTSYS6	PVMG1	Defines the connection to N2N6AVSC through VTAM

Table 12, Table 14 on page 88, and Figure 27, illustrate a sample PVMG configuration that is defined in this chapter. You may want to create similar tables and diagrams to help you configure your PVM/PVMG network .

```
LANG AMENG
OPID OPERATOR
PVMPROP OPERATOR
LOCAL PVMG1 N2ECAPV2
LINK * RMTPVM1 IUCV USER PVM
LOGONID N2N6AVSC
SELECT RMTPVM2
* Route Table:
ROUTE DEFAULT  RMTPVM1
AUTHORIZ OPERATOR *
SESSDISC N2ECAVSC
SESSTERM ####
XDBC CONS
*N2N6AVSC
LU N2N6AVSC N2ECAP01-20 RESTRICT
* APPLID TABLE
APPLN RMTSYS6 N2N6AVSC
* Logon Mode Table Entry Name Keywords and Values:
LOGMODE 3277    S3270
LOGMODE 3278-2B D4B32782
LOGMODE 3278-2E D4E32782
LOGMODE 3278-3B D4B32783
LOGMODE 3278-3E D4E32783
LOGMODE 3278-4B D4B32784
LOGMODE 3278-4E D4E32784
LOGMODE 3278-5B D4B32785
LOGMODE 3278-5E D4E32785
LOGMODE 3279-2B D4B32792
LOGMODE 3279-2E D4E32792
LOGMODE 3279-3B D4B32793
LOGMODE 3279-3E D4E32793
LOGMODE 3290    D329001
DUMP VMDUMP OPERATOR
MSGOPTNS Y Y Y Y Y Y Y Y
```

Figure 27. Sample Configuration File for PVMG1

Table 12. Configuration File Planning Table Number 3. Defines the PVMG link to the local PVM.

Line Address	Node ID	Driver Type	User	VM ID	Comments
*	RMTPVM1	IUCV	USER	PVM	This is the link to the local PVM node.

The following tables contain detailed information about the records in the sample PVMG configuration file above.

Table 13. PVMG Configuration File Planning Table Number 4. Defines the VTAM APPLID to which the PVM terminals log on.

APPLID	APPLICATION	Comments
VTAM	ASC	See the LOGONID configuration record

Table 14. PVMG Configuration File Planning Table Number 5. Defines the Routing from PVMG Nodes to Target Nodes

Target ID	Next ID	Comments
DEFAULT	RMTPVM1	See the ROUTE configuration record

Table 15 (Page 1 of 2). PVMG Configuration File Planning Table Number 6. Specifying the VTAM Logon Mode Table Entries Used by PVMG

Entry Description	Entry Name	Comments
3277	S3270	3270 BSC / 3270 SDLC / 3270 LOCAL NON-SNA
3278-2B	D4B32782	3274 MODEL 1B/1D WITH MODEL 2 SCREEN (LOCAL NON-SNA) 3274 MODEL 1C BSC WITH MODEL 2 SCREEN 3276 BSC WITH MODEL 2 SCREEN PRIMARY SCREEN 24X80 (1920) NO ALTERNATE SCREEN DEFINED
3278-2E	D4E32782	3274 MODEL 1B/1D WITH MODEL 2 SCREEN (LOCAL NON-SNA) 3274 MODEL 1C BSC WITH MODEL 2 SCREEN - WITH EDS 3276 BSC WITH MODEL 2 SCREEN PRIMARY SCREEN 24X80 (1920) NO ALTERNATE SCREEN DEFINED
3278-3B	D4B32783	3274 MODEL 1B/1D WITH MODEL 3 SCREEN (LOCAL NON-SNA) 3274 MODEL 1C BSC WITH MODEL 3 SCREEN 3276 BSC WITH MODEL 3 SCREEN PRIMARY SCREEN 24X80 (1920) ALTERNATE SCREEN 32X80 (2560)
3278-3E	D4E32783	3274 MODEL 1B/1D WITH MODEL 3 SCREEN (LOCAL NON-SNA) 3274 MODEL 1C BSC WITH MODEL 3 SCREEN - WITH EDS 3276 BSC WITH MODEL 3 SCREEN PRIMARY SCREEN 24X80 (1920) ALTERNATE SCREEN 32X80 (2560)
3278-4B	D4B32784	3274 MODEL 1B/1D WITH MODEL 4 SCREEN (LOCAL NON-SNA) 3274 MODEL 1C BSC WITH MODEL 4 SCREEN 3276 BSC WITH MODEL 4 SCREEN PRIMARY SCREEN 24X80 (1920) ALTERNATE SCREEN 43X80 (3440)
3278-4E	D4E32784	3274 MODEL 1B/1D WITH MODEL 4 SCREEN (LOCAL NON-SNA) 3274 MODEL 1C BSC WITH MODEL 4 SCREEN - WITH EDS 3276 BSC WITH MODEL 4 SCREEN PRIMARY SCREEN 24X80 (1920) ALTERNATE SCREEN 43X80 (3440)
3278-5B	D4B32785	3274 MODEL 1B/1D WITH MODEL 5 SCREEN (LOCAL NON-SNA) 3274 MODEL 1C BSC WITH MODEL 5 SCREEN 3276 BSC WITH MODEL 5 SCREEN PRIMARY SCREEN 24X80 (1920) ALTERNATE SCREEN 27X132 (3564)
3278-5E	D4E32785	3274 MODEL 1B/1D WITH MODEL 5 SCREEN (LOCAL NON-SNA) 3274 MODEL 1C BSC WITH MODEL 5 SCREEN - WITH EDS 3276 BSC WITH MODEL 5 SCREEN PRIMARY SCREEN 24X80 (1920) ALTERNATE SCREEN 27X132 (3564)

Table 15 (Page 2 of 2). PVMG Configuration File Planning Table Number 6. Specifying the VTAM Logon Mode Table Entries Used by PVMG

Entry Description	Entry Name	Comments
3279-2B	D4B32792	3279 MODEL 2B - NO EDS PRIMARY SCREEN 24X80 (1920) ALTERNATE SCREEN 24X80 (1920)
3279-2E	D4E32792	3279 MODEL 2B - WITH EDS PRIMARY SCREEN 24X80 (1920) ALTERNATE SCREEN 24X80 (1920)
3278-3B	D4B32793	3279 MODEL 3 - NO EDS PRIMARY SCREEN 24X80 (1920) ALTERNATE SCREEN 32X80 (2560)
3279-3E	D4E32793	3279 MODEL 3 - WITH EDS PRIMARY SCREEN 24X80 (1920) ALTERNATE SCREEN 32X80 (2560)
3290	D329001	LOGMODE TABLE ENTRY FOR 3290 TERMINAL PRIMARY SCREEN SIZE 24X80 ALTERNATE SCREEN SIZE 62X160

Commenting the PVMG Configuration File

Use the comment (*) record to put comments in the configuration file. For example,

```
* Configuration file for node PVMG1
```

is the first record of the configuration file for PVMG1. The asterisk (*) must be in column 1.

Comment records can be placed anywhere in the configuration file.

Identifying the System Language

Use the LANG record to identify the default national language for a node. For example,

```
LANG AMENG
```

assigns American English as the default national language for PVMG1.

The LANG record should be placed at the top of the PVMG configuration file.

Excluding Double-Byte Character Messages

Use the XDBC record to exclude users from receiving DBCS messages. For example, if the LANG record identifies a double-byte character language such as KANJI and the PVMG console should not receive DBCS message, enter the following:

```
XDBC CONS
```

Now the PVMG console receives any DBCS messages in the backup language.

See "QUERY" on page 318 for information on the DBCS settings and "CDBC" on page 289 for information on changing DBCS-exclusion settings.

Identifying the Local Node Name and the VTAM Application Name

Use the LOCAL record to identify the local PVMG node name. For example,

```
LOCAL PVMG1
```

assigns the node ID (PVMG1) to the local PVMG system.

```
LOCAL PVMG1 N2ECAPV2
```

assigns a node ID (PVMG1) and VTAM application name (N2ECAPV2) to the local PVMG system.

Defining a Link from PVMG to PVM

Use the LINK record to define a link for connecting with the PVM virtual machine. LINK records must follow the LOCAL records in the configuration file. The following records define a LINK from node RMTPVM1.

```
LINK * RMTPVM1 IUCV USER PVM
```

is the link defined from node PVMG1 to RMTPVM1.

Note: If the link on the PVM side is not active, PVMG will automatically retry the IUCV connect at 60 seconds intervals until either communications is established, or a PVMG DROP node command is issued by an operator.

Defining the VTAM APPLID to Which Users Log on

Use the LOGONID record to define the VTAM APPLID to which the PVMG node is to log the PVM terminals if they access a target application using the PVMG node ID.

```
LOGONID N2N6AVSC
```

Assigning Node Names to PF Keys

Use the SELECT record to assign frequently used node names to PF keys. Your installation can define up to eleven PF keys in the PVMG configuration file.

The first SELECT record *nodeid* is assigned to PF1, the second SELECT record *nodeid* is assigned to PF2, and so forth to PF7. The eighth SELECT record is assigned to PF9 (PF8 is reserved for the SCROLL FORWARD function). The ninth SELECT record is assigned to PF10, and so forth to PF12.

If you use a SELECT record without specifying the node ID, the PF key associated with that SELECT record is omitted. This allows a PF key to be skipped.

```
SELECT RMTPVM2
SELECT RMTPVM3
SELECT RMTPVM4
```

Defining Routes and Alternate Routes

Use the route record to define the routing path from the local PVMG node to a specified target node. For PVMG to PVM communications, this route must be consistently defined on both ends of network paths.

Use the second form of the route command to define default routing for those nodes not defined by LINK or ROUTE records. You can only enter one default route record.

```
ROUTE DEFAULT RMTPVM1
```

Specifying LU Names Used by PVMG with Secure VTAM APPLIDs

To create a secure LU session to a VTAM application, specify a nodename and user ID on the LU record (see "LU" on page 408).

An example of a secure LU would be:

```
LU vtamapplid RMTPVM1 SMITH
```

Only user ID SMITH at RMTPVM1 would have access to *vtamapplid*.

To create a range of restricted LU sessions to the VTAM network, specify RESTRICT on the LU record.

An example of a range of restricted LUs would be:

```
LU N2N6AVSC N2ECAP01-20 RESTRICT
```

This causes 20 logical units to be generated. Only one session may be assigned to a LU.

Authorizing PVMG Operators

Use the AUTHORIZ record to identify users allowed to enter the restricted subset of PVMG commands and the nodes from which these commands may be issued. For example,

```
AUTHORIZ OPR
AUTHORIZ JONES RMTPVM1
```

authorizes the person identified by the user ID OPR to enter restricted PVMG commands from the local node, PVMG1. The second AUTHORIZ statement authorizes user JONES at RMTPVM1 to enter restricted PVMG commands.

Saving PVMG Information

The DUMP record saves information about PVMG. Use the DUMP record to specify the type of dump to be taken if abnormal termination of PVMG occurs. For example, the record:

```
DUMP VMDUMP
```

causes a VMDUMP dump to be taken of all PVMG storage. This type of dump must be specified if you process dumps using IPCS, or the VM/ESA and VM/XA Dump Viewing Facility. The record:

```
DUMP CP JONES
```

causes a CP DUMP to be taken of all PVMG storage, and sent to user ID JONES.

Specifying Message Requirements

You can use the MSGNOH and the MSGOPTNS records to indicate how and where you want messages displayed. Use the MSGNOH record to indicate that responses to all commands entered by the CP SMSG command must be returned using the CP MSGNOH command rather than the CP MSG command.

If you enter more than one MSGNOH record in the PVMG configuration file, only the last MSGNOH record in the file is accepted. The other MSGNOH records are ignored.

```
MSGNOH
```

Use the MSGOPTNS record to determine if the time stamps or message numbers for messages will be directed to the PVMG user ID, the users, and/or the operator. For example, the record:

```
MSGOPTNS Y Y
```

tells PVMG to specify the time stamps and the message numbers on the messages to the PVMG user ID.

Specifying the Target User ID for Natural Language Message Reception

Use the OPID record to specify the virtual machine ID that receives messages that usually go to the system operator. You may also use this record to stop sending messages to the system operator. For example, the following record forward all messages to the JONES user ID:

```
OPID JONES
```

The following example halts messages from being sent to a CP, or alternate, operator.

```
OPID NONE
```

Specifying the User ID of the Programmed Operator Virtual Machine That Receives Encoded Messages

Use the PVMPROP record to identify a virtual machine that receives a copy of PVMG error messages. All messages are transmitted to the virtual machine by the CP Special Message (SMSG) facility. The messages are sent to the PVMPROP virtual machine in the same format as those sent to the PVM Command Response Interface (CRI). For example, the following records would cause all PVMG messages to be sent in encoded form by the CP SMSG command to the PROP user ID:

```
PVMPROP PROP
```

Specifying the VTAM APPLID to Disconnect When PVMG Fails

Use the SESSDISC record to indicate the VTAM APPLID from which user sessions should be disconnected (rather than stop the sessions) when an unexpected PVMG problem occurs.

```
SESSDISC N2ECAVSC
```

will disconnect the APPLID when an unexpected PVMG problem occurs.

Changing the Default Local PVMG Node Session Termination String

Use the SESSTERM record to change the default session termination string used at the local PVMG node.

For example, to set the session terminate string to ####, specify:

```
SESSTERM ####
```

Specifying VTAM Logon Model Table Entries Used by PVMG

Use the LOGMODE record to specify VTAM logmode table entries that are used by PVMG when establishing interactive sessions.

```
*  Logon Mode Table Entry Name Keywords and Values:
LOGMODE 3277    S3270
LOGMODE 3278-2B D4B32782
LOGMODE 3278-2E D4E32782
LOGMODE 3278-3B D4B32783
LOGMODE 3278-3E D4E32783
LOGMODE 3278-4B D4B32784
LOGMODE 3278-4E D4E32784
LOGMODE 3278-5B D4B32785
LOGMODE 3278-5E D4E32785
LOGMODE 3279-2B D4B32792
LOGMODE 3279-2E D4E32792
LOGMODE 3279-3B D4B32793
LOGMODE 3279-3E D4E32793
LOGMODE 3290    D329001
```

Tailoring the PROFILE GCS File

A PROFILE GCS file is needed to IPL GCS on the PVMG server virtual machine.

You can tailor the PROFILE GCSSAMP file that was installed when you installed PVMG. Use the following sample file as a guide.

```
/* */
'CP SET RUN OFF'
'FILEDEF PVMGCONFIG DISK PVMG CONFIG *'
'CP SET IMSG ON'
'GLOBAL LOADLIB PVMG'
'LOADCMD PVMG EFGMAN'
'PVMG INIT'
```

Figure 28. Sample PVMG Profile GCS

Chapter 8. Developing NAMES Files for MPVM

After MPVM is installed on your VM system, you must set up the MPVMSYS NAMES file to reflect the nicknames and characteristics of the sessions your installation will be using. The MPVMSYS NAMES file contains entries, and each entry consists of a nickname and subtags. This chapter gives you details on what your system names file should contain.

Note: You should rename the MPVMSYS NAMESAMP file to MPVMSYS NAMES and use it as sample NAMES file.

Also, an MPVM user can create a personal NAMES file, called MPVM NAMES. Users might need their own file, for instance, when they want to specify nicknames that are not in the MPVMSYS NAMES file, or when they want to override entries defined in MPVMSYS NAMES. The end of this chapter contains more detail about user's name files.

Use the MPVMSYS NAMES file, which should now be included on the MPVM common disk for PVM 2.1, as a base for creating session entries. You need to define entries for target applications and special entries for the overall MPVM environment. The following table describes the sample files supplied on the PVM 2.1.1 LOCALSAM (2C2) disk for MPVM users:

Table 16. PVM 2.1.1 Supplied Sample Files for MPVM users

File Name	Sample Description
MPVM NAMESAMP	Sample user level MPVM names file.
MPVMP SNTENTRY	Sample 370 feature SNT entry.
MPVMSGNS NAMESAMP	Sample system level MPVM Auto-Signon names file.
MPVMSYS NAMESAMP	Sample system level MPVM names file.

Here is an overview of what is contained in this chapter:

- Developing MPVMSYS NAMES file entries that define PVM target applications
- Developing MPVMSYS NAMES file entries that define configuration information for MPVM
- Special, miscellaneous MPVMSYS NAMES file entries
- Verifying your completed MPVM installation
- How individual MPVM NAMES files relate to the system-level MPVMSYS NAMES file.

Plus, starting on page 99, there is a sample MPVMSYS NAMES file that summarizes many of the tags you can use.

Creating Entries for Applications

MPVM gets its instructions from the values assigned to entries in the MPVMSYS NAMES file. Each target application must have a nickname; specified by the :nick. tag. You must specify the characteristics about the nickname using subtags under the :nick..

© Copyright IBM Corp. 1991,1993

Specifying the Nickname

When users want to access applications through PVM, they specify a nickname from either the MPVM selection screen or the MPVM command or macro. Each nickname entry, specified by the :nick. tag, defines the target application to be accessed.

You should create a nickname entry for each application that you want MPVM users to access. The value specified by the nickname:

- Must be unique
- Can be up to eight characters
- Must contain only the characters a-z, A-Z, 0-9, or one of the following special characters:
  . , _ : % & @ # ?
- Cannot be the word *CANCEL*
- Cannot start with an escape (default value ?) character

For example, if your application updates customer billing information, you might have a nickname of *BILLING*. The nickname entry in the MPVMSYS NAMES file would look like:

```
:nick.BILLING
```

Even if your nicknames contain lower case characters, they are converted to uppercase during processing.

Specifying Characteristics for Application Nicknames

For MPVM to establish a session with this target application, however, it needs to know more than just a user's nickname. MPVM must have specific session characteristics. PVM gets this information from values you associate with the nickname. The values correspond to MPVM variables.

How do you set these variables in a NAMES file? With subtags that look the same as the MPVM variable name. We will describe some examples in the next few sections.

Defining Target Applications

Use the ***:target.*** subtag in the NAMES file to specify the name of a target system. So, for example, if VMSYS1 is the name of a target system, you could specify a subtag/value pair like:

```
:target.VMSYS1
```

:target. must be specified as:

- :target subtag in a nickname entry
- allowing it to default to the :target subtag in the ?default nickname
- allowing it to default to the nickname.

Defining Other Variables for Applications

There are several other MPVM variables that you may want to specify for an application nickname. Any MPVM variable that can be set (with a SET command, or directly on the MPVM command or macro) can be specified on a NAMES file subtag. Some of the most common examples are PF key and PA key settings. See the *VM/Pass-Through Facility Programming Reference* for information on all MPVM variables.

Specifying Commands to be Invoked

Besides specifying a subtag/value pair that corresponds to setting an MPVM variable, you can also specify a string that you want MPVM to directly start. You do this using a special **:?** tag. This can be used, for example, to have MPVM process a macro when it processes a particular nickname. MPVM processes the command on this tag completely before moving onto the next tag in the NAMES file. For example, `:?.MSG HI THERE!` sends the message `HI THERE!` to the nickname being processed.

Specifying Your Own Subtags

While MPVM is processing an application nickname, it ignores any subtags that begin with a character other than A-Z, a-z, * or ?. This lets you create your own subtags. For example, you could create a subtag called **:$DATE.** that specifies when a particular entry was added to the MPVMSYS NAMES file. The MPVM NICKINFO command could be used to search the MPVMSYS NAMES file for this subtag.

Specifying Nickname Lists

An individual nickname tag could specify the name of a *list* of nicknames rather than contain subtag/value pairs. The list name must be prefix by a * character on the :nick. tag. The :nick. tag must be followed by a special :* subtag, which contains the individual nicknames in the list.

For example, suppose you had nicknames PROFS, LOCAL1, and CUSTADR for three of your applications. You could define a single list name called ALLSYS and specify the NAMES file entry like this:

```
:nick.*ALLSYS
   :*.PROFS LOCAL1 CUSTADR
```

When you select *ALLSYS, you select all three applications.

Summary of Application NAMES File Entries

Here is a summary of the steps you must do to create a NAMES file for MPVM:

1. XEDIT the sample NAMES file for MPVM. (As distributed, this file was named MPVMSYS NAMESAMP; during installation we renamed it to MPVMSYS NAMES.)
2. Determine all the applications you want users to access through MPVM, and where those applications are located.

3. For each application listed in step 2 on page 97, do the following steps:
   a. Enter these two mandatory tags with associated values:

Tag	Description
:nick.	A nickname for the application that users will specify, either from the MPVM Session Selection Menu, or as an operand on the MPVM command or macro.
:target.	The target system name associated with this session.

   b. Look through the MPVM variable list and determine what other characteristics you want to assign to the MPVM session.
4. Place the MPVMSYS NAMES file on the MPVM common disk, and verify that all entries for remote applications work. Each entry nickname should be separately checked to verify that it works properly.

You can have up to 999 nickname entries in an MPVMSYS or MPVM NAMES file, and up to 999 subtags per nickname entry.

Defining Configuration Settings—:nick.?CONFIG

The MPVMSYS NAMES file does more than just define information for application nicknames—it includes some special nickname values that define important MPVM characteristics. ?CONFIG is a special value on the nickname tag (**:nick.?CONFIG**) that is processed during MPVM initialization.

User Initialization Settings

Use the following subtag to define the user ID of the PVM virtual machine:

:pvmid.

The subtag can be in upper, lower, or mixed case. If you omit this subtag, the user ID of the PVM virtual machine defaults to *PVM*.

Creating Other Special Entries

MPVM allows other special nicknames that start with a question mark (?).

Setting Global Values for All Sessions—:nick.?INITIAL

?INITIAL is a special value on the :nick. tag that is processed only once, when MPVM is first started, before any sessions are established. You can specify subtags for ?INITIAL to set values for all sessions. However, you can only set global variables, which are common to all sessions. You can also specify a **:?.** tag to process a given string once, before any sessions are established.

Setting Default Values for All Sessions—:nick.?DEFAULT

?DEFAULT is a special value on the :nick. tag that is processed whenever a new session is established, before an individual system nickname is processed. ?DEFAULT can be used to set local *and* global default values. You can also specify a **:?.** tag to process a given string before any new session is established.

Cleaning Up After the Last Session Has Ended—:nick.?FINAL

?FINAL is a special value on the :nick. tag that is processed whenever the last MPVM session has ended, before the main MPVM module (EFGVIEW) terminates. ?FINAL can be used to undo some things originally set up when establishing your MPVM environment (perhaps on the **:?INITIAL** tag). The subtag typically used on ?FINAL is the **:?**.

Sample MPVMSYS NAMES File

The following page shows an example of typical nickname entries and subtags you would have in an MPVMSYS NAMES file.

```
* The ?CONFIG entry contains the PVM virtual machine user ID
:nick.?CONFIG
    :pvmid.PVM
*
*
* The ?INITIAL entry is processed when MPVM is first invoked, providing
*  global settings for all MPVM sessions.  This example defines the SYSREQ key
*  to process the SELECT command (to roll to the next session).
:nick.?INITIAL
    :sysreq.SELECT
*
* The ?DEFAULT entry is processed when each new MPVM session is established.
*  This example defines the PF10 key to display the previously-issued command,
* and defines the default target as RMTPVM1.
:nick.?DEFAULT
    :pf10.RETRIEVE
    :target.RMTPVM1
    :nick.?FINAL
    :?.MSG YOU ARE LEAVING MPVM
* APPLICATION DEFINITIONS
*
* This defines a session on the local system.
* The MSG command is processed directly, displaying the indicated message.
:nick.LOCAL1
   :target.RMTPVM1
   :?.MSG This text gets displayed
*
* This example defines a session called PROFS; the PROFS* target application is on a system
*  named WESTVM1.
:nick.PROFS
   :target.WESTVM1
*
* This example defines a session called CUSTADDR; the CUSTADDR target application is on a
*  system named MVSSYST in an SNA network.
*  This example also shows how you can remove the MPVM
*  definition for PF10 so that the target application's PF10 definition is used.

:nick.CUSTADDR
   :target.MVSSYST
   :pf10.
*
* This example shows how to specify a nickname list:  A user issuing MPVM *ALLSYS
*  automatically gets 3 sessions established.
:nick.*ALLSYS
   :*.PROFS LOCAL1 CUSTADDR
```

Figure 29. Sample MPVMSYS NAMES File

Verifying the MPVMSYS NAMES File

After creating your MPVMSYS NAMES file, you need to verify that it works properly.

Log onto another user ID and enter:

```
MPVM
```

You should see an MPVM Selection Screen. Try each nickname entry to verify that session characteristics are working as you intended.

Considerations for Individual User NAMES Files

We mentioned that users can also have a personal NAMES file to define information about their MPVM applications. When MPVM resolves a nickname, the user-level NAMES file (if it exists) is checked first. If the user-level file does not contain the specified nickname, CMS searches the system-level NAMES file. If a nickname is defined in both the MPVM NAMES file and the MPVMSYS NAMES file, the information in the MPVM NAMES file takes precedence.

A user's NAMES file can include the special ?INITIAL, ?DEFAULT, and ?FINAL nickname entries. Subtags here are added to any subtags in the ?INITIAL, ?DEFAULT, and ?FINAL entries in MPVMSYS NAMES.

User files can have their own ?CONFIG entry and subtags. The only subtag that MPVM will look at, however, is :pvmid.. All other ?CONFIG subtags are ignored.

The MPVM Session Selection Menu includes a convenient feature for users working on their own NAMES file: If you go to the command line of the menu and edit your NAMES file (`?x mpvm names`), then add a nickname entry and file it, the new nickname dynamically appears on the menu, in its proper order.

As an alternative to specifying subtag/value pairs in the MPVM NAMES file, users can also specify them:

- As operands on the MPVM command or macro call
- From the command line on the MPVM Session Selection Menu.

Summary of MPVMSYS NAMES Contents

The following table summarizes all the tags that can be used in MPVM's system-level NAMES file, called MPVMSYS NAMES:

Table 17. Partial Contents of the MPVMSYS NAMES File

Nickname	Usage and Subtags
:**nick.***appl_nickname*	This is a typical nickname value. *appl_nickname* is a nickname you set up for the application or system that users want to establish sessions with. *:varname.value.* lets you to set values for each session. The subtag corresponds directly to an MPVM variable name. Some examples of varaibles that can be set as a subtag are: :target. Node ID for the system MPVM is accessing. :route. PVM secondary route. :port. PVM port number for sessions through a node that emulates a 327x controller. :cinit. user data for sessions connected through SNA. :logmode. SNA logmode table name that the PVMG server should use. :PF*nn*. command string processed when the PF*nn* key is pressed. :columns. number of columns for the session screen. :rows. number of rows for the session screen. :signon. command string processed during an installation-written automatic signon routine. :initcmd command string processed when a session with a target system begins. **Note:** All local variables that are settable by the MPVM user can be used as subtags in this context. See the *VM/Pass-Through Facility User's Guide* for a complete list of MPVM variables. *:?.executable_string* is a special tag telling MPVM to run the string as it is processed. This can also be used with special ?INITIAL and ?DEFAULT nicknames, and with ?FINAL. **:$***anystring.anymeaning* lets you to create your own tags. Actually, the first character does not have to be a dollar sign; MPVM ignores any character other than A-Z, a-z, or ?.
:**nick.****listname*	*listname* is a special nickname that points to a list of other nicknames. :**nick.****listname* must be followed by the **:*.** subtag, in this format: *:*.appl_nickname1 appl_nickname2 ...*
:**nick.?CONFIG**	This special nickname value is processed once, when MPVM is first started. ?CONFIG has one special, reserved subtag: **:pvmid.** defines the user ID of the virtual machine containing the PVM server.
:**nick.?INITIAL**	This special nickname value is processed once, when MPVM is first invoked. ?INITIAL subtags can be used to set values for all sessions. Only global variables can be set.
:**nick.?DEFAULT**	This special nickname value is processed whenever a new session is established, before an individual system nickname is processed, and can be used to specify local and global default values.
:**nick.?FINAL**	This special nickname value is processed whenever the last MPVM session is terminated and MPVM is getting ready to end. This can be used for some final *clean up* activity.

Chapter 9. Operating the Network

In this chapter, you will find a discussion of the PVM commands that are used to operate the PVM network. Operating includes tasks such as starting, querying, and stopping the PVM network, quiescing, resuming, and dropping devices and users, and communicating with the network's users.

If you are interested in automating the operations of your PVM network, see Chapter 15, "Facilities for Automating PVM and PVMG Operations" on page 251.

The following is based on the network defined in Chapter 6, "Defining Your PVM Configuration." The chapter shows sample uses of commands and the output or messages that appear at the PVM operator's display station. All commands are fully described in Appendix C, "PVM Commands" and Appendix E, "PVMG Commands."

Note: All samples shown are for PVM 2.1.1. Messages may vary for PVM 2.1. The messages may not appear on your screen as the samples shown in this book.

Starting PVM

Once the network is defined in the configuration file and the PVM profile exec and PROFILE EXEC are created, you can enter the RUNPVM command. The RUNPVM command loads and starts the PVM module, which uses the configuration file to define the system setup. Once PVM is up, the PVM profile exec is called. The RUNPVM command can be started from your CMS PROFILE EXEC, or you can enter it from a CMS command line.

If you use the default names for the configuration file (PVM CONFIG), the PVM profile exec (PROFILE PVM), and the PVM module (PVM MODULE), you load and start the PVM module by issuing:

```
runpvm
```

However, in this scenario, to process RMTPVM1 CONFIG (the configuration file for RMTPVM1) and RMTPVM1 PVM (the PVM profile exec for RMTPVM1) and to load and start the PVM module, enter

```
runpvm * rmtpvm1 rmtpvm1
```

© Copyright IBM Corp. 1991,1993

You will receive messages similar to the following:

```
DMSLIO740I Execution begins...
03/31/93 09:41:38 DVMBFR040I VM Pass-Through Facility 2.1.1000, built 03/30/93
15:27:08, is ready
03/31/93 09:41:40 DVMCOS001I Your EXEC command is complete
03/31/93 09:41:40 DVMCON050I ************************************************
03/31/93 09:41:40 DVMCON050I * PVM Profile Exec for node RMTPVM1 - RMTPVM1 PVM *
03/31/93 09:41:40 DVMCON050I ************************************************
03/31/93 09:41:40 DVMCON050I START LINE 080
03/31/93 09:41:40 DVMCON050I START LINE 081
03/31/93 09:41:40 DVMCON050I START LINE 082
03/31/93 09:41:40 DVMCON050I START LINE 02F
03/31/93 09:41:40 DVMCON050I START LINE 0B0
03/31/93 09:41:40 DVMCON050I START LINE 200
03/31/93 09:41:40 DVMCON050I START LINE 201
03/31/93 09:41:40 DVMCON050I START LINE 500
03/31/93 09:41:40 DVMCON050I START NODE RMTPVM5B
03/31/93 09:41:40 DVMCON050I START NODE PVMG1
03/31/93 09:41:40 DVMCON050I START GRAF 450
03/31/93 09:41:40 DVMCON050I START GRAF 451
03/31/93 09:41:40 DVMCON050I START GRAF 452
03/31/93 09:41:40 DVMCON050I START GRAF 453
03/31/93 09:41:40 DVMCON050I START GRAF 454
03/31/93 09:41:40 DVMCON050I START PRT 491 VSEAPL
03/31/93 09:41:40 DVMCON068I GRAF 450 DEFINED
03/31/93 09:41:40 DVMNET318I GRAF450 is starting
03/31/93 09:41:40 DVMCON068I GRAF 451 DEFINED
03/31/93 09:41:40 DVMNET318I GRAF451 is starting
03/31/93 09:41:40 DVMCON068I GRAF 452 DEFINED
03/31/93 09:41:40 DVMNET318I GRAF452 is starting
03/31/93 09:41:40 DVMCON068I GRAF 453 DEFINED
03/31/93 09:41:40 DVMNET318I GRAF453 is starting
03/31/93 09:41:40 DVMCON068I GRAF 454 DEFINED
03/31/93 09:41:40 DVMNET318I GRAF454 is starting
03/31/93 09:41:40 DVMCON068I DEVICE RESET
03/31/93 09:41:40 DVMCON068I DEVICE RESET
03/31/93 09:41:40 DVMCON068I DEVICE RESET
03/31/93 09:41:40 DVMCON068I DEVICE RESET
03/31/93 09:41:40 DVMCON068I DEVICE RESET
```

Now, suppose RMTPVM1 CONFIG contains errors. After issuing the following command,

```
runpvm * rmtpvm1 rmtpvm1
```

you will receive messages similar to the following:

```
DMSLIO740I Execution begins...
DVMDIR014E The LINK address is not valid
DVMDIR010E LINK 300 VSEAPL 327X
DVMDIR018E The previous LINK type was not 327X
DVMDIR010E PORT 0 3278-2 RMTPVM1 USERA
DVMDIR018E The previous LINK type was not 327X
DVMDIR010E PORT 1 3278-2 RMTPVM1 0351
DVMDIR018E The previous LINK type was not 327X
DVMDIR010E PORT 8 3278-2 RMTPVM1 USERB
DVMDIR018E The previous LINK type was not 327X
DVMDIR010E PORT 9 328X RMTPVM1 0360
DVMDIR012E The configuration record is out of order
DVMDIR010E AUTHORIZ OPR
DVMDIR012E The configuration record is out of order
DVMDIR010E AUTHORIZ JONES RMTPVM2
DVMDIR821E SELECT node is not valid
DVMDIR010E SELECT VSEAPL
03/31/93 10:26:20 DVMBFR040I VM Pass-Through Facility 2.1.1000, built 03/30/93
15:45:18, is ready
03/31/93 10:26:22 DVMCOS001I Your EXEC command is complete
03/31/93 10:26:22 DVMCON050I ************************************************
03/31/93 10:26:22 DVMCON050I * PVM Profile Exec for node RMTPVM1 - RMTPVM1 PVM *
03/31/93 10:26:22 DVMCON050I ************************************************
03/31/93 10:26:22 DVMCON050I START LINE 081
03/31/93 10:26:22 DVMCON050I START LINE 082
03/31/93 10:26:22 DVMCON050I START LINE 02F
03/31/93 10:26:22 DVMCON050I START LINE 0B0
03/31/93 10:26:22 DVMCON050I START LINE 200
03/31/93 10:26:22 DVMCON050I START LINE 201
03/31/93 10:26:22 DVMCON050I START LINE 300
03/31/93 10:26:22 DVMCON052E Invalid address "300" on START LINE
03/31/93 10:26:22 DVMCON050I START LINE 500
03/31/93 10:26:22 DVMCON050I START NODE RMTPVM5B
03/31/93 10:26:22 DVMCON050I START NODE PVMG1
03/31/93 10:26:22 DVMCON050I START GRAF 450
03/31/93 10:26:22 DVMCON050I START GRAF 451
03/31/93 10:26:22 DVMCON050I START GRAF 452
03/31/93 10:26:22 DVMCON050I START GRAF 453
03/31/93 10:26:22 DVMCON050I START GRAF 454
03/31/93 10:26:22 DVMCON050I START PRT 491 VSEAPL
03/31/93 10:26:22 DVMCON068I GRAF 450 DEFINED
03/31/93 10:26:22 DVMNET318I GRAF450 is starting
03/31/93 10:26:22 DVMCON068I GRAF 451 DEFINED
03/31/93 10:26:22 DVMNET318I GRAF451 is starting
03/31/93 10:26:22 DVMCON068I GRAF 452 DEFINED
03/31/93 10:26:22 DVMNET318I GRAF452 is starting
03/31/93 10:26:22 DVMCON068I GRAF 453 DEFINED
03/31/93 10:26:22 DVMNET318I GRAF453 is starting
03/31/93 10:26:22 DVMCON068I GRAF 454 DEFINED
03/31/93 10:26:22 DVMNET318I GRAF454 is starting
03/31/93 10:26:22 DVMCON068I DEVICE RESET
03/31/93 10:26:22 DVMCON068I DEVICE RESET
03/31/93 10:26:22 DVMCON068I DEVICE RESET
03/31/93 10:26:22 DVMCON068I DEVICE RESET
03/31/93 10:26:22 DVMCON068I DEVICE RESET
```

The following list contains the errors in the configuration file and corrections that need to be made.

- The LINK address is not valid

The address on LINK 300 VSEAPL 327X is not valid. The address for a 327x device must range from X'20' to X'FF'. Change the LINK record to LINK 080 VSEAPL 327X.

- The previous LINK type was not 327X

  Since the address on the LINK record is not valid, all the PORT records associated with that LINK record produce error statements. Once the address on the LINK record is corrected, these error statements do not appear.

- The configuration record is out of order

  The AUTHORIZ OPR and AUTHORIZ JONES RMTPVM2 records are out of order. The configuration file records were arranged as follows:

```
⋮
ROUTE DEFAULT RMTPVM3
AUDIT DATA RECORD
* The configuration file records to this point are required to
* define all routes from RMTPVM1 to the other nodes.
* The following records might apply to any PVM node.
AUTHORIZ OPR
AUTHORIZ JONES RMTPVM2
SUBSET LISTFILE
LNODE * AUTO
⋮
```

All AUTHORIZ records must appear before the AUDIT DATA RECORD in the configuration file. Move the AUTHORIZ OPR and AUTHORIZ JONES RMTPVM2 before the AUDIT DATA RECORD record. The configuration file records should be arranged as follows:

```
⋮
ROUTE DEFAULT RMTPVM3
* The configuration file records to this point are required to
* define all routes from RMTPVM1 to the other nodes.
* The following records might apply to any PVM node.
AUTHORIZ OPR
AUTHORIZ JONES RMTPVM2
SUBSET LISTFILE
AUDIT DATA RECORD
LNODE * AUTO
⋮
```

- SELECT node is not valid

  The SELECT VSEAPL record is not valid since the address on the LINK record for node ID VSEAPL was not valid.

The following list contains the error statements from your PVM profile exec. These error statements, however, are corrected when the errors in the configuration file are corrected.

- Invalid address 300 on START LINE

  Since address 300 was not valid in the LINK record in the configuration file, the START LINE 300 record could not execute.

Querying the System

As a PVM operator, you will want to determine the status of the system, lines, devices, and users.

Suppose you want to see the information about the system, such as, the number of links on the system, the link names, group names, or the number of users on the system. Enter:

```
q system
```

You will receive messages similar to the following:

```
03/31/93 09:43:41 DVMCON050I Q SYSTEM
03/31/93 09:43:41 DVMQRY080I The local node is RMTPVM1, Users 0, PVM 2.1.1000,
built 03/30/93 15:27:08
03/31/93 09:43:41 DVMSIT081I Link VSEAPL   Type 327X    Address  080     Users
 0 DOWN
03/31/93 09:43:41 DVMSIT081I Link MVSTSO   Type 327X    Address  081     Users
 0 DOWN
03/31/93 09:43:41 DVMQRY081I Link RMTPVM2A Type CTCA    Address  200     Users
 0 CONNECT  Group RMTPVM2
03/31/93 09:43:41 DVMQRY081I Link RMTPVM2B Type CTCA    Address  201     Users
 0 CONNECT  Group RMTPVM2
03/31/93 09:43:41 DVMC39081I Link RMTPVM5A Type 3088    Address  500/501 Users
 0 DOWN     Group RMTPVM5
03/31/93 09:43:41 DVMAPE081I Link RMTPVM5B Type APPC    Resource EAGLE   Users
 0 DOWN     Group RMTPVM5
03/31/93 09:43:41 DVMQRY081I Link RMTPVM3  Type BSCA    Address  082     Users
 0 DOWN
03/31/93 09:43:42 DVMQRY081I Link RMTSYS2  Type ROCF    Address  02F     Users
 0 DOWN
03/31/93 09:43:42 DVMRMH081I Link REM3270  Type R3270   Address  0B0     Users
 0 DOWN
03/31/93 09:43:43 DVMVTC081I Link PVMG1 Type IUCV Path 0 Users 0 ACTIVE
```

These messages tell you all the defined links on the local PVM node RMTPVM1, the group names, the type of each link, the address of each link, the number of users on each link, the status of each link (only the CTCA links, line 200 and 201 forming group RMTPVM2, are connected), the number of users connected to RMTPVM1 (0), the level of PVM Version 2, and the time and date the PVM system was built.

Or, you may just need information about a specific line. Then, enter:

```
q line 200
```

You will receive messages similar to the following:

```
03/31/93 09:44:16 DVMCON050I Q LINE 200
03/31/93 09:44:16 DVMQRY081I Link RMTPVM2A Type CTCA    Address  200     Users
 0 CONNECT  Group RMTPVM2
```

These messages tell you the same information as `q system` except it is just for the specified line.

Suppose a user dials into RMTPVM1, you will receive messages similar to the following:

```
03/31/93 09:44:50 DVMUGR150I User GRAF450 connected to node RMTPVM1  Address WG7
51308
03/31/93 09:44:50 DVMAPP180I Logical device L031 created for RMTPVM1 GRAF450, ad
dress WG751308
```

These messages tell you that a device whose ID is GRAF450 and whose address is WG751308 (VTAM logical unit name) is connected to RMTPVM1 at logical address L031. If you want additional information about this display device, enter:

```
q user graf450
```

You will receive messages similar to the following:

```
03/31/93 09:45:03 DVMCON050I Q USER GRAF450
03/31/93 09:45:03 DVMQRY088I RMTPVM1 GRAF450 Device WG751308 3278 Model 2 H is
in session at RMTPVM1 L031, Bytes In 1101 Out 134
```

These messages tell you that device GRAF450 is a 3278 Model 2 display station.

Now, suppose two other users enter the DIAL command—one dials directly into RMTPVM1 and the other dials into RMTPVM1 through RMTPVM2. You may want to display the status of each active link and each user session. To do this, enter:

```
q sys q
```

```
03/31/93 10:11:45 DVMCON050I Q SYS Q
03/31/93 10:11:45 DVMQRY080I The local node is RMTPVM1, Users 2, PVM 2.1.1000,
built 03/30/93 15:27:08
03/31/93 10:11:45 DVMQRY088I RMTPVM1 GRAF451 Device 017E 3279 Model 3 HPC is in
session at RMTPVM1 L03B, Bytes In 1101 Out 200
03/31/93 10:11:45 DVMQRY088I RMTPVM1 GRAF450 Device WG751308 3278 Model 2 H is
in session at RMTPVM1 L031, Bytes In 1101 Out 134
03/31/93 10:11:45 DVMQRY081I Link RMTPVM2A Type CTCA     Address  200      Users
 1 CONNECT  Group RMTPVM2
03/31/93 10:11:45 DVMQRY088I RMTPVM2 GRAF460 Device 0D4D 3277 Model 2  is in
session at RMTPVM1 L03C, Bytes In 1095 Out 0
03/31/93 10:11:46 DVMQRY081I Link RMTPVM2B Type CTCA     Address  201      Users
 0 CONNECT  Group RMTPVM2
```

Now, suppose a user, whose ID is SMITH, enters the PASSTHRU command to get to RMTPVM1. You will receive messages similar to the following:

```
03/31/93 10:13:21 DVMAPP180I Logical device L02B created for RMTPVM1 SMITH, addr
ess WG752308
03/31/93 10:13:21 DVMUIN150I User SMITH connected to node RMTPVM1 L02B Address W
G752308
```

To display additional information about SMITH, enter:

```
q user smith
```

You will receive messages similar to the following:

```
03/31/93 10:13:32 DVMCON050I Q USER SMITH
03/31/93 10:13:32 DVMQRY088I RMTPVM1 SMITH Device WG752308 3278 Model 2 H is in
session at RMTPVM1 L02B, Bytes In 1101 Out 134
```

These messages tell you that user SMITH is connected to RMTPVM1 from a 3278 Model 2 display station.

Now to see the status of the system again, enter:

```
q system
```

You will receive messages similar to the following:

```
03/31/93 10:14:08 DVMCON050I Q SYSTEM
03/31/93 10:14:08 DVMQRY080I The local node is RMTPVM1, Users 5, PVM 2.1.1000,
built 03/30/93 15:27:08
03/31/93 10:14:08 DVMSIT081I Link VSEAPL   Type 327X     Address  080
Users  0 DOWN
03/31/93 10:14:08 DVMSIT081I Link MVSTSO   Type 327X     Address  081
Users  0 DOWN
03/31/93 10:14:08 DVMQRY081I Link RMTPVM2A Type CTCA     Address  200
Users  1 CONNECT  Group RMTPVM2
03/31/93 10:14:08 DVMQRY081I Link RMTPVM2B Type CTCA     Address  201
Users  0 CONNECT  Group RMTPVM2
03/31/93 10:14:08 DVMC39081I Link RMTPVM5A Type 3088     Address  500/501
Users  0 DOWN     Group RMTPVM5
03/31/93 10:14:08 DVMAPE081I Link RMTPVM5B Type APPC     Resource EAGLE
Users  0 DOWN     Group RMTPVM5
03/31/93 10:14:08 DVMQRY081I Link RMTPVM3  Type BSCA     Address  082
Users  0 DOWN
03/31/93 10:14:08 DVMQRY081I Link RMTSYS2  Type ROCF     Address  02F
Users  0 DOWN
03/31/93 10:14:08 DVMRMH081I Link REM3270  Type R3270    Address  0B0
Users  0 DOWN
03/31/93 10:14:39 DVMVTC081I Link PVMG1 Type IUCV Path 0 Users 0 ACTIVE
```

Defining a Link Dynamically

Suppose you need to add a new link to a group, and you do not want to shutdown PVM, then enter:

```
define rmtpvm2c new type ctca line 202 group rmtpvm2
```

You will receive messages similar to the following:

```
03/31/93 10:14:47 DVMCON050I DEFINE RMTPVM2C NEW TYPE CTCA LINE 202 GROUP RMTPVM
2
03/31/93 10:14:47 DVMCON001I Your DEFINE command is complete
```

To display the status of each active link and each user session, enter:

```
q sys q
```

If user SMITH2 has entered MPVM and started two sessions, query information for those sessions will also be displayed.

You will receive messages similar to the following:

```
03/31/93 10:16:28 DVMCON050I Q SYS Q
03/31/93 10:16:28 DVMQRY080I The local node is RMTPVM1, Users 5, PVM 2.1.1000,
built 03/30/93 15:27:08
03/31/93 10:16:28 DVMQRY088I RMTPVM1 SMITH Device WG752308 3278 Model 2 H is in
session at RMTPVM1 L02B, Bytes In 1101 Out 134
03/31/93 10:16:28 DVMMPG088I RMTPVM1 SMITH2 *SES0002 Device L014F 3278
Model 2 HC is in session at RMTPVM1 L182, Bytes In 1718 Out 110
03/31/93 10:16:28 DVMMPG088I RMTPVM1 SMITH2 *SES0001 Device L014F 3278
Model 2 HC is in session at RMTPVM1 L181, Bytes In 1718 Out 110
03/31/93 10:16:28 DVMQRY088I RMTPVM1 GRAF451 Device 017E 3279 Model 3 HPC is in
session at RMTPVM1 L03B, Bytes In 1101 Out 200
03/31/93 10:16:28 DVMQRY088I RMTPVM1 GRAF450 Device WG751308 3278 Model 2 H is
in session at RMTPVM1 L031, Bytes In 1101 Out 134
03/31/93 10:16:28 DVMQRY081I Link RMTPVM2A Type CTCA     Address  200      Users
 1 CONNECT  Group RMTPVM2
03/31/93 10:16:28 DVMQRY088I RMTPVM2 GRAF460 Device 0D4D 3277 Model 2  is in
session at RMTPVM1 L03C, Bytes In 1095 Out 0
03/31/93 10:16:28 DVMQRY081I Link RMTPVM2B Type CTCA     Address  201      Users
 0 CONNECT  Group RMTPVM2
03/31/93 10:16:28 DVMQRY081I Link RMTPVM2C Type CTCA     Address  202      Users
 0 CONNECT  Group RMTPVM2
```

Now you have added a new CTCA link to group RMTPVM2 from RMTPVM1 to RMTPVM2.

Displaying the Path of a Session

Use the SHOW command to display the path of a particular session. For example,

```
show path rmtpvm2
```

shows the path PVM took to get to RMTPVM2. You will receive messages similar to the following:

```
03/31/93 10:17:07 DVMCON050I SHOW PATH RMTPVM2
03/31/93 10:17:08 DVMCOR922I The path to node RMTPVM2 is through node RMTPVM1
03/31/93 10:17:08 DVMNEV127E RMTPVM2 RMTPVM3 LINK IS DOWN
FROM RMTPVM3: Your SHOW command is complete
```

Sending Messages

There are three commands you can use to send users messages. The commands are LMSG, MSG, and PMSG. Use the LMSG message to send messages to all PVM users associated with the local PVM node or a specified line or group. For example,

```
lmsg rmtpvm1 Are you coming to Monday's meeting?
```

sends the message to all PVM users connected to RMTPVM1. You will receive messages similar to the following:

```
03/31/93 10:17:34 DVMCON050I LMSG RMTPVM1 ARE YOU COMING TO MONDAY'S MEETING?
03/31/93 10:17:34 DVMCOU001I Your LMSG command is complete
```

Use the MSG command to send messages to a remote station or to a specified VM user at a specified node. For example,

```
msg rmtpvm2 hayes Are you coming to Tuesday's meeting?
```

sends the message to HAYES at RMTPVM2. You will receive messages similar to the following:

```
03/31/93 10:19:42 DVMCON050I MSG RMTPVM2 HAYES ARE YOU COMING TO TUESDAY'S MEETI
NG?
```

Use the PMSG command to send messages to all PVM users connected to the local PVM node. For example,

```
pmsg all Are you coming to Wednesday's meeting?
```

sends the message to all PVM users connected to RMTPVM1. You will receive messages similar to the following:

```
03/31/93 10:20:39 DVMCON050I PMSG ALL ARE YOU COMING TO WEDNESDAY'S MEETING?
03/31/93 10:20:39 DVMCOU001I Your PMSG command is complete
```

Issuing CMS Subset Mode Commands

Since the configuration file contains the record `SUBSET LISTFILE`, you can display the CMS files on your A-disk by entering:

```
listfile
```

If the configuration file did not contain the `SUBSET LISTFILE` record, you would have to prefix the LISTFILE command with *CMS* to process the command.

Quiescing, Resuming, and Dropping Links, Devices, and Users

To quiesce line 200, enter:

```
quiesce line 200
```

You will receive messages similar to the following:

```
03/31/93 10:22:18 DVMCON050I QUIESCE LINE 200
03/31/93 10:22:18 DVMCON001I Your QUIESCE command is complete
```

To check the status of the active links on RMTPVM1, enter:

```
q sys q
```

You will receive messages similar to the following:

```
03/31/93 10:22:28 DVMCON050I Q SYS Q
03/31/93 10:22:28 DVMQRY080I The local node is RMTPVM1, Users 5, PVM 2.1.1000,
built 03/30/93 15:27:08
03/31/93 10:22:28 DVMQRY088I RMTPVM1 SMITH Device WG752308 3278 Model 2 H is in
session at RMTPVM1 L02B, Bytes In 1101 Out 134
03/31/93 10:22:28 DVMMPG088I RMTPVM1 SMITH2 *SES0002 Device L014F 3278
Model 2 HC is in session at RMTPVM1 L182, Bytes In 1718 Out 110
03/31/93 10:22:28 DVMMPG088I RMTPVM1 SMITH2 *SES0001 Device L014F 3278
Model 2 HC is in session at RMTPVM1 L181, Bytes In 1718 Out 110
03/31/93 10:22:28 DVMQRY088I RMTPVM1 GRAF451 Device 017E 3279 Model 3 HPC is in
session at RMTPVM1 L03B, Bytes In 1101 Out 200
03/31/93 10:22:28 DVMQRY088I RMTPVM1 GRAF450 Device WG751308 3278 Model 2 H is
in session at RMTPVM1 L031, Bytes In 1101 Out 134
03/31/93 10:22:28 DVMQRY081I Link RMTPVM2A Type CTCA     Address  200      Users
 1 QUIESCE  Group RMTPVM2
03/31/93 10:22:28 DVMQRY088I RMTPVM2 GRAF460 Device 0D4D 3277 Model 2  is in ses
sion at RMTPVM1 L03C, Bytes In 1400 Out 12
03/31/93 10:22:28 DVMQRY081I Link RMTPVM2B Type CTCA     Address  201      Users
 0 CONNECT  Group RMTPVM2
03/31/93 10:22:28 DVMQRY081I Link RMTPVM2C Type CTCA     Address  202      Users
 0 CONNECT  Group RMTPVM2
```

Since there is a user connected on line 200, the line is put in quiesce state rather than being dropped. If the user session ends, then line 200 would be dropped.

However, if the user session does not end and you want line 200 *connected* again, enter:

```
resume line 200
```

You can see the system is back to the same status (line 200 is CONNECTed) as it was before issuing the QUIESCE command by entering:

```
q sys q
```

You will receive message similar to the following:

```
03/31/93 10:22:38 DVMCON050I Q SYS Q
03/31/93 10:22:38 DVMQRY080I The local node is RMTPVM1, Users 5,
03/31/93 10:22:38 DVMQRY088I RMTPVM1 SMITH Device WG752308 3278 Model
2 H is in session at RMTPVM1 L02B, Bytes In 1101 Out 134
03/31/93 10:22:38 DVMMPG088I RMTPVM1 SMITH2 *SES0002 Device L014F 3278
Model 2 HC is in session at RMTPVM1 L182, Bytes In 1718 Out 110
03/31/93 10:22:38 DVMMPG088I RMTPVM1 SMITH2 *SES0001 Device L014F 3278
Model 2 HC is in session at RMTPVM1 L181, Bytes In 1718 Out 110
03/31/93 10:22:38 DVMQRY088I RMTPVM1 GRAF451 Device 017E 3279 Model
3 HPC is in session at RMTPVM1 L03B, Bytes In 1101 Out 200
03/31/93 10:22:38 DVMQRY088I RMTPVM1 GRAF450 Device WG751308 3278
Model 2 H is in session at RMTPVM1 L031, Bytes In 1101 Out 134
03/31/93 10:22:38 DVMQRY081I Link RMTPVM2A Type CTCA     Address  200
Users  1 CONNECT  Group RMTPVM2
03/31/93 10:22:38 DVMQRY088I RMTPVM2 GRAF460 Device 0D4D 3277 Model
2  is in session at RMTPVM1 L03C, Bytes In 1400 Out 12
03/31/93 10:22:39 DVMQRY081I Link RMTPVM2B Type CTCA     Address  201
Users  0 CONNECT  Group RMTPVM2
03/31/93 10:22:39 DVMQRY081I Link RMTPVM2C Type CTCA     Address  202
Users  0 CONNECT  Group RMTPVM2
03/31/93 10:22:39 DVMVTC081I Link PVMG1 Type IUCV Path 0 Users 0 ACTIVE
```

To place the display device GRAF450 into quiesce mode, enter:

```
quiesce graf 450
```

You will receive messages similar to the following:

```
03/31/93 10:22:52 DVMCON050I QUIESCE GRAF 450
03/31/93 10:22:52 DVMCON001I Your QUIESCE command is complete
```

The user who dialed into RMTPVM1 can complete the session. However, once the user session is over, the user is dropped and the display support ends.

After issuing QUIESCE, if you do not want the user dropped or the display support ended after the user session completes, enter

```
resume graf 450
```

You will receive messages similar to the following:

```
03/31/93 10:23:16 DVMCON050I RESUME GRAF 450
03/31/93 10:23:16 DVMCON001I Your RESUME command is complete
```

Now, the display device GRAF450 remains available even if the user session completes.

If you want to end the display support for device GRAF450, enter

```
drop graf 450
```

You will receive messages similar to the following:

```
03/31/93 10:23:30 DVMCON050I DROP GRAF 450
03/31/93 10:23:30 DVMCON001I Your DROP command is complete
```

Any active user session on display GRAF450 is also ended.

If you want to end SMITH user's session, enter

```
drop user rmtpvm1 smith
```

You will receive messages similar to the following:

```
03/31/93 10:24:03 DVMCON050I DROP USER RMTPVM1 SMITH
03/31/93 10:24:03 DVMCON001I Your DROP command is complete
03/31/93 10:24:03 DVMAPP181I Logical device L02B ended for RMTPVM1 SMITH, addres
s WG752308,
03/31/93 10:24:03 DVMAPP181I  Bytes In 134 Out 1101
03/31/93 10:24:03 DVMUIN151I User SMITH dropped from node RMTPVM1 L02B address W
G752308,
03/31/93 10:24:03 DVMUIN151I  Bytes In 1101 Out 134
```

SMITH's session at L02B is ended. Then SMITH is dropped from RMTPVM1.

Suppose you want to drop line 200 connecting RMTPVM1 to RMTPVM2. Then, enter:

```
drop line 200
```

You will receive messages similar to the following:

```
03/31/93 10:24:23 DVMCON050I DROP LINE 200
03/31/93 10:24:23 DVMCON001I Your DROP command is complete
03/31/93 10:24:23 DVMCTC501I Link RMTPVM2A Blocks In 11 Out 10, Bytes In 692 Out
 2285, Delay 189, Average 68
03/31/93 10:24:23 DVMCTC314I Link RMTPVM2A is terminating via command
03/31/93 10:24:23 DVMNEU105W Link RMTPVM2A Line 0200 Status is DOWN
```

These messages tell you that line 200 to RMTPVM2 is down.

Using the PVM Programmable Operator (PVMPROP)/Command Response Interface (CRI)

Using the PVMPROP programmable operator and the PVM Command Response Interface, you can write applications that monitor PVM operations. When an problem or error occurs, PVM can send the message to the PVMPROP user ID using the CP SMSG command. An application program running on the PVMPROP user ID can act as a PVM operator and respond to the message, or send instructions to a real PVM operator.

For more information about PVMPROP and the Command Response Interface, see Chapter 15, "Facilities for Automating PVM and PVMG Operations."

Stopping PVM

There are two ways you can stop the PVM system. You could use the SHUTDOWN command or the QUIT command. The QUIT command stops the system immediately without notifying the users. Therefore, before issuing the QUIT command, you should first try the SHUTDOWN command with the IMMED option. When using this SHUTDOWN command, users receive messages informing them that their session are being terminated.

To stop PVM, enter:

```
shutdown immed
```

You will receive messages similar to the following:

```
03/31/93 10:24:51 DVMCON050I SHUTDOWN IMMED
03/31/93 10:24:51 DVMNET313I PVM system shutdown has started
03/31/93 10:24:51 DVMUGR319I Task GRAF454 is terminating
GRAF 454  DETACHED
03/31/93 10:24:51 DVMCON068I GRAF 454  DETACHED
03/31/93 10:24:53 DVMUGR319I Task GRAF453 is terminating
GRAF 453  DETACHED
03/31/93 10:24:53 DVMCON068I GRAF 453  DETACHED
03/31/93 10:24:53 DVMUGR319I Task GRAF452 is terminating
GRAF 452  DETACHED
03/31/93 10:24:53 DVMCON068I GRAF 452  DETACHED
03/31/93 10:24:53 DVMCTC501I Link RMTPVM2B Blocks In 8 Out 10, Bytes In 564 Out
654, Delay 191, Average 67
03/31/93 10:24:54 DVMCTC314I Link RMTPVM2B is terminating via command
03/31/93 10:24:54 DVMNEU105W Link RMTPVM2B Line 0201 Status is DOWN
03/31/93 10:24:54 DVMCTC501I Link RMTPVM2C Blocks In 4 Out 4, Bytes In 240 Out 2
40, Delay 192, Average 59
03/31/93 10:24:54 DVMCTC314I Link RMTPVM2C is terminating via command
03/31/93 10:24:54 DVMNEU105W Link RMTPVM2C Line 0202 Status is DOWN
```

Starting and Stopping Printer Sessions

PVM supports printer sessions in two different ways, depending on the way the printer device is attached to the system. The two **types** of printer sessions are:

Type I Using 3271/3274/3275 emulation, PVM supports printer sessions between 3270 Information Display system printers and non-PVM target systems. The system printers for this type of session are owned by CP. Either local or remote attachments to CP are allowed. You can assign any 3271/3274/3275 logical port to a directly attached 3270 Information Display System printer. You define printer ports in the configuration file using the PORT record.

Type II You can initiate sessions for printers that are part of a remote cluster managed by PVM. Type II printers are not owned by CP. The target node ID for these remote printers can be either PVM or non-PVM. Remote printers on remote clusters are defined with CLPORT statements in the configuration file.

Printer Session Initiation

Type I Printer Session

PVM supports printer sessions between 3270 Information Display system printers and non-PVM target systems (accessed by 3271/3274 emulation). The subject printers must have been attached to the PVM system by the CP ATTACH command (local printer) or NETWORK ATTACH (remote printer), or by the DEDICATE control statement in the CP directory.

Printer sessions, unlike interactive sessions, are typically controlled by an installation not by individual users. The printer session is initiated by the restricted START PRT command. The command may be included in an EXEC like PROFILE PVM or may be entered in the command stream.

The START PRT command specifies the target non-PVM node for the printer session and, optionally, a logical port number associated with the printer. If no port is specified, PVM attempts to satisfy the request by using a port defined (configuration file) for the subject printer or an unreserved port on the associated emulator line. It is advisable to assign specific printers to specific ports to ensure that the printer expected by the target application is the desired printer, and matches the actual printer device type and characteristics.

The START PRT command activates a session between the specified printer and the target non-PVM system. Once the printer session is established, users of a target application may direct output to the printer device through its associated port. For printers in this type of session, the extended features (extended color, extended highlighting, and programmed symbols) of the 3270 Information Display System printers are not supported.

Output is initiated from the target application by the user. By using procedures established by that application, users may direct output to the emulator printer port. The target application must manage the transfer of data to that port. Scheduling of 328x printer output is under the control of the PVM installation and the target application. Procedures for use and management of this resource are the responsibility of the customer.

Type II Printer Session

As mentioned in the previous section, remote clusters provide another type of printer session. An R3270 line driver attempts to start a session with the node (node ID) on the CLPORT record. This is done for either PVM or non-PVM target nodes. If the target is a PVM node, a logical device is created at the target to represent the cluster printer. If the CLPORT record specifies it, the remote line driver also attempts to attach the logical device to a virtual machine ID as a specific virtual address. For example, if this CLPORT record appeared in PVM configuration file:

```
CLPORT 06 3286 RMTPVM1 L100 RSCS 0C0
```

then a 3286 printer would be logically connected to RSCS. PVM creates a printer logical device, at a logical address (using the address offset value of 100) on the local VM system. The local system, in this example, is at node RMTPVM1. PVM issues a CP ATTACH command to attach the device to RSCS at address 0C0. All output directed to the logical printer by RSCS will be forwarded by PVM to the real printer.

The START LINE command can be used to start display or printer line sessions. A line driver task is created to handle the remote line. The line driver determines the status of the line and clusters and devices attached to the line.

If the printer is not ready, at the time START processing takes place, (or at the time the control unit becomes available) then the printer is still started. Once the session is initiated, it will be the responsibility of the application target to cause the user to manually ready the printer.

The DROP USER command can be entered to restart the printer session because of an unforeseen problem at the user ID. This causes the printer session to be terminated, and a new session established as described above.

Note: When a printer which supports extended data streams is connected to a control unit which supports extended data streams, the line driver will wait until the printer is readied to start a remote printer session. Extended data streams

support can then be passed along at initiation to the target. This support includes extended color, programmed symbol sets, and extended highlighting.

If a logical device address of L000 is specified, the next available logical printer device in the PVM server machine will be used for this printer session. Therefore, logical device address 000 cannot be requested as a printer device.

Printer Session Termination

Type I Printer Session

A printer session created by the START PRT command can be terminated by either a DROP PRT command, which causes immediate termination of the session, or by a QUIESCE PRT command, which terminates the printer session whenever there has been no I/O activity for a 2-minute interval.

Type II Printer Session

The DROP LINE command is used to end an R3270 line driver. However, because there may be more than one display or printer on a line, the DROP LINE command should not be used with the intention of terminating an individual remote display or printer session. You can use the DROP USER command to recycle the printer session.

Starting PVMG

Once PVMG is installed and added to GCS, starting PVMG is simply a matter of IPLing GCS, or issuing RUNPVMG from your PROFILE EXEC.

Querying PVMG

As a PVMG operator, you will want to determine the status of the system, sessions, nodes, and users. The sections that follow provide examples of how to use the QUERY command.

Querying the System

Suppose you want to see the information about the system, such as, the number of sessions on the system, line status, or the number of users on the system. Enter:

```
pvmg q system
```

You will receive messages similar to the following:

```
03/31/93 22:20:53 EFGOQS080I VM Pass-Through Gateway Services 2.1.1000,
Users 0, Node PVMG1, built 03/31/93 22:16:25
03/31/93 22:20:53 EFGOCP081I Link RMTPVM1 Type IUCV Users 0 CONNECT
```

These messages tell you all the defined links on the local PVMG node PVMG1, the number of users on each link, the level of PVMG Version 2 and the time and date the PVMG system was built.

Monitoring User Activity

When a PVM user accesses an SNA application, such as N2N6AVSC, a message similar to the following displays:

```
03/31/93 22:21:43 EFGUSG102I Session start from RMTPVM1 SMITH to PVMG1 GRAF0001,
Application N2NBASAM, Logical Unit N2ECAP01
```

You may want information about a specific PVM user. Then, enter:

```
pvmg q user rmtpvm1 smith
```

You will receive messages similar to the following:

```
03/31/93 22:21:50 EFGOCP088I RMTPVM1 SMITH Device 1D9 3279 Model 2 HC is
in session at PVMG1 GRAF0001
03/31/93 22:21:50 EFGOCP088I Application N2NBASAM, Logical Unit N2ECAP01,
Bytes In 1701 Out 110
```

These messages tell you the same information as `q system` except it is just for the specified user.

Suppose an SNA user (from SNA application AL3E0Q) accesses a PVM node, such as RMTPVM5, a message similar to the following displays:

```
03/31/93 22:21:59 EFGUSG102I Session start from PVMG1 AL3E0Q to RMTPVM5
L120
```

Displaying the Path of a PVMG Session

Use the SHOW command to display the path of a particular session. For example,

```
pvmg show path rmtpvm2
```

shows the path from the PVMG node to the RMTPVM2. You will receive messages similar to the following:

```
03/31/93 22:22:04 EFGOCP922I The path to node RMTPVM1 is through node PVMG1
```

Sending Messages

There are three commands you can use to send users messages. The commands are LMSG, MSG, and PMSG. Use the LMSG message to send messages to all users whose sessions access PVMG through VTAM, or whose sessions access VTAM through PVMG. For example,

```
pvmg lmsg rmtpvm1 Are you coming to Monday's meeting?
```

sends the message to all users connected to RMTPVM1.

Use the MSG command to send messages to a remote station or to a specified VM user ID at a specified node. For example,

```
pvmg msg rmtpvm2 hayes Are you coming to Tuesday's meeting?
```

sends the message to HAYES at RMTPVM2.

Use the PMSG command to send messages to all PVMG users connected to the local PVMG node. For example,

```
pvmg pmsg all Are you coming to Wednesday's meeting?
```

sends the message to all PVMG users connected to PVMG1.

Dropping Users and Nodes

To quiesce the link to node RMTPVM1 enter:

```
pvmg quiesce node RMTPVM1
```

Now, to check the status of the active links on PVMG1, enter:

```
pvmg q sys q
```

You will receive messages similar to the following:

```
03/31/93 22:22:29 EFGOQS080I VM Pass-Through Gateway Services 2.1.1000,
Users 2, Node PVMG1, built 03/31/93 22:16:25
03/31/93 22:21:50 EFGOCP088I RMTPVM1 SMITH Device 1D9 3279 Model 2 HC is
in session at PVMG1 GRAF0001
03/31/93 22:21:50 EFGOCP088I Application N2NBASAM, Logical Unit N2ECAP01,
Bytes In 1701 Out 110
03/31/93 22:22:29 EFGO05088I PVMG1 AL3E0R Device AL3E0R 3278 Model 2 H
is in session at RMTPVM5 L120, Bytes In 1423 Out 1964
03/31/93 22:22:29 EFGOCP081  Link RMTPVM1 Type IUCV Users 2 Quiesce
```

Since there are users connected on RMTPVM1, the link is put in quiesce state rather than being dropped. Once the users' sessions end, then link RMTPVM1 would be dropped.

However, if the user session does not end and you want link RMTPVM1 *connected* again, enter:

```
pvmg resume node RMTPVM1
```

You can see the system is back to the same status (RMTPVM1 is CONNECTed) as it was before issuing the QUIESCE command by entering:

```
pvmg q sys q
```

You will receive messages similar to the following:

```
03/31/93 22:23:29 EFGOQS080I VM Pass-Through Gateway Services 2.1.1000,
Users 2, Node PVMG1, built 03/31/93 22:16:25
03/31/93 22:21:50 EFGOCP088I RMTPVM1 SMITH Device 1D9 3279 Model 2 HC is
in session at PVMG1 GRAF0001
03/31/93 22:21:50 EFGOCP088I Application N2NBASAM, Logical Unit N2ECAP01,
Bytes In 1701 Out 110
03/31/93 22:23:29 EFGO05088I PVMG1 AL3E0R Device AL3E0R 3278 Model 2 H
is in session at RMTPVM5 L120, Bytes In 1423 Out 1964
03/31/93 22:23:29 EFGOCP081  Link RMTPVM1 Type IUCV Users 2 Connect
```

You can quiesce the local PVMG server by entering:

```
pvmg quiesce system
```

Any users with sessions through PVMG1 can complete the session. However, once the users' sessions are over, the users are dropped.

After issuing QUIESCE, if you do not want the users dropped after the user sessions complete, enter

```
pvmg resume system
```

The PVMG1 server remains available even if the user's sessions complete.

If you want to end the session for user SMITH, enter:

```
pvmg drop user pvmg1 graf0001
```

You will receive messages similar to the following:

```
03/31/93 22:26:03 EFGUST103I Session ended from RMTPVM1 SMITH to PVMG1 GRAF0001,
Application N2NBASAM, Logical Unit N2ECAP01
03/31/93 22:26:03 EFGOQS001I Your DROP command is complete
```

SMITH's SNA session ends, and is dropped from link RMTPVM1.

If you want to end the server support for link RMTPVM1, enter

```
pvmg drop RMTPVM1
```

You will receive messages similar to the following:

```
03/31/93 22:25:30 EFGCCT105W Link RMTPVM1 Status is DOWN
```

Stopping PVMG

Issue the SHUTDOWN command. If there are no active sessions, the PVMG server will shutdown immediately. If there are active sessions, the operator can choose to wait for the sessions to end, or can enter the following command:

```
pvmg shutdown immed
```

Users will receive messages informing them that their session will be terminated. The PVMG server then shuts down.

You will receive messages similar to the following:

```
03/31/93 22:30:23 EFGOED3131 PVMG System Shutdown has Started
03/31/93 22:30:23 EFGOPE626I The PVMG command processor has terminated
03/31/93 22:30:23 EFGCTC626I The PVMG link control task has terminated
03/31/93 22:30:23 EFGUSH626I The PVMG VTAM session control task has terminated
03/31/93 22:30:23 EFGMAI100I PVMG is terminating
```

Note: Shutdown without the IMMED option allows active sessions to continue, but will not allow new sessions to start. When the last session ends, shutdown of the PVMG server completes.

Chapter 10. Using CVIEW

This chapter offers information that can help you with the daily operation and administration of the CVIEW virtual machine.

Starting the CVIEW Virtual Machine

You can start the CVIEW virtual machine either automatically or manually. Once the CVIEW virtual machine is started, the consult and conference sessions are available for use.

Automatically Starting CVIEW

To automatically start the CVIEW virtual machine from another virtual machine, ensure that the:

- VM/ESA directory contains an AUTOLOG1 or equivalent user ID.
- AUTOLOG1 virtual machine has Privilege Class A or B and this command (with no variable data) is specified in its PROFILE EXEC:

  ```
  CP AUTOLOG CVIEW PASSWORD
  ```

- CVIEW virtual machine directory contains an IPL CMS entry
- CVIEW PROFILE EXEC contains an EXEC DXGCVIEW command.

For a complete description of the CP AUTOLOG command, see the *VM/ESA: CP Command and Utility Reference*, SC24-5519.

Once the CVIEW virtual machine is automatically logged on, it runs in a disconnected mode.

Manually Starting CVIEW

You can start the CVIEW virtual machine manually using these steps:

1. Clear the VM/ESA system logo from the screen.
2. Enter:

   ```
   L CVIEW
   ```

3. Enter the password when prompted.
4. If an IPL CMS was not specified in the directory for the CVIEW virtual machine, enter:

   ```
   IPL CMS
   ```

5. Enter a null line to run the PROFILE EXEC.
6. If an EXEC DXGCVIEW command was not specified in the PROFILE EXEC for the CVIEW virtual machine, when the IPL completes enter:

   ```
   DXGCVIEW
   ```

7. After CVIEW is running, you can enter:

   ```
   CP DISC
   ```

   to disconnect the CVIEW virtual machine.

Once CVIEW is running, users can enter the DIAL CVIEW command from their own display stations to start consult and conference sessions.

© Copyright IBM Corp. 1991,1993

CVIEW Commands

CVIEW has both *general* and *privileged* commands. The general commands can be used by any CVIEW user to obtain operating status and information about the CVIEW virtual machine. Users defined as privileged (during customization) and users logged on to the CVIEW virtual machine console can use the privileged commands. Privileged commands administer and control the operation of the CVIEW virtual machine.

The commands that follow can be entered from the CVIEW virtual machine console. However, if you want to enter commands from a virtual machine console other than the CVIEW virtual machine console use the CP SMSG command. For example, a general user who wants to enter the CVIEW HELP command from his virtual machine console enters:

```
SMSG userid HELP
```

where *userid* is the user ID of the CVIEW virtual machine that will receive the command. (HELP is a CVIEW general command).

For a complete description of the CP SMSG command, see the *VM/ESA: CP Command and Utility Reference*, SC24-5519.

CMS

Purpose

Lets you enter CMS commands within the CVIEW virtual machine.

Authority

Privileged

Operands

command
Is the CMS command to be processed.

EXEC *command*
Specifies that the command to be processed is an exec. *command* is the file name of the EXEC.

Usage Notes

1. The CVIEW virtual machine does not perform other functions while a CMS command is being processed.
2. CMS commands that process slowly can effect users engaged in CVIEW sessions.

CP

Purpose

Lets you enter CP commands within the CVIEW virtual machine.

Authority

Privileged

Operands

command
Is the CP command to be processed.

Usage Notes

1. The CP LOGOFF command is not detected by CVIEW; if entered, it will log off the CVIEW virtual machine.

DROP

Purpose

Stops an active GRAF device by number or a CVIEW session by number, name, or logical device address. The CVIEW session ends if the:

- GRAF address is assigned to the moderator's display station.
- Session has only two users.

Authority

Privileged

Operands

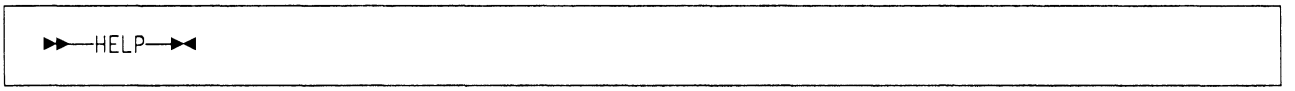
GRAF *nnn*
Indicates the 3270 display station that has dialed the CVIEW virtual machine. The *nnn* is the virtual address assigned by the CVIEW virtual machine to a 3270 display station. Valid GRAF addresses are 501 through 530. Use QUERY GRAF to find the active GRAF address for a display station.

SESSION *xxxxxxxx*
Indicates an active consult or conference session. The *xxxxxxxx* is an unlisted consultant's identification (up to eight alphanumeric characters), a

conference name (up to eight alphanumeric characters) or a listed consultant's ID (a number from 1 to 12).

LDEV L*nnn*
Indicates a consult or conference session by the logical device address assigned by CP for communications between CP and a CVIEW session. L*nnn* is the LDEV address assigned by CP to a CVIEW session. Use the QUERY ACTIVE command to obtain the LDEV address.

HELP



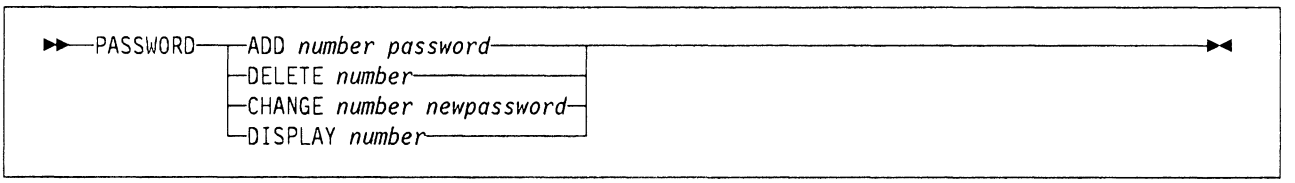
Purpose

Displays on-line information about CVIEW commands. Privileged users receive help information for all privileged and general commands. Other users receive help information for the general commands.

Authority

General

PASSWORD



Purpose

Maintains passwords for consultants that are listed in the CONSULT IDNAME file and appear on the GIVE and RECEIVE panels. You can also maintain consultant passwords by editing the CONSULT IDNAME file.

Authority

Privileged

Operands

ADD
Assigns a password to a consultant with no current password.

number
Is a value from 2 to 12 that corresponds to the consultant in the CONSULT IDNAME file.

password
Is the password to be assigned to the specified consultant. The password can be up to eight characters long.

DELETE
Removes a consultant's password from the CONSULT IDNAME file.

CHANGE
Assigns a different password to a consultant.

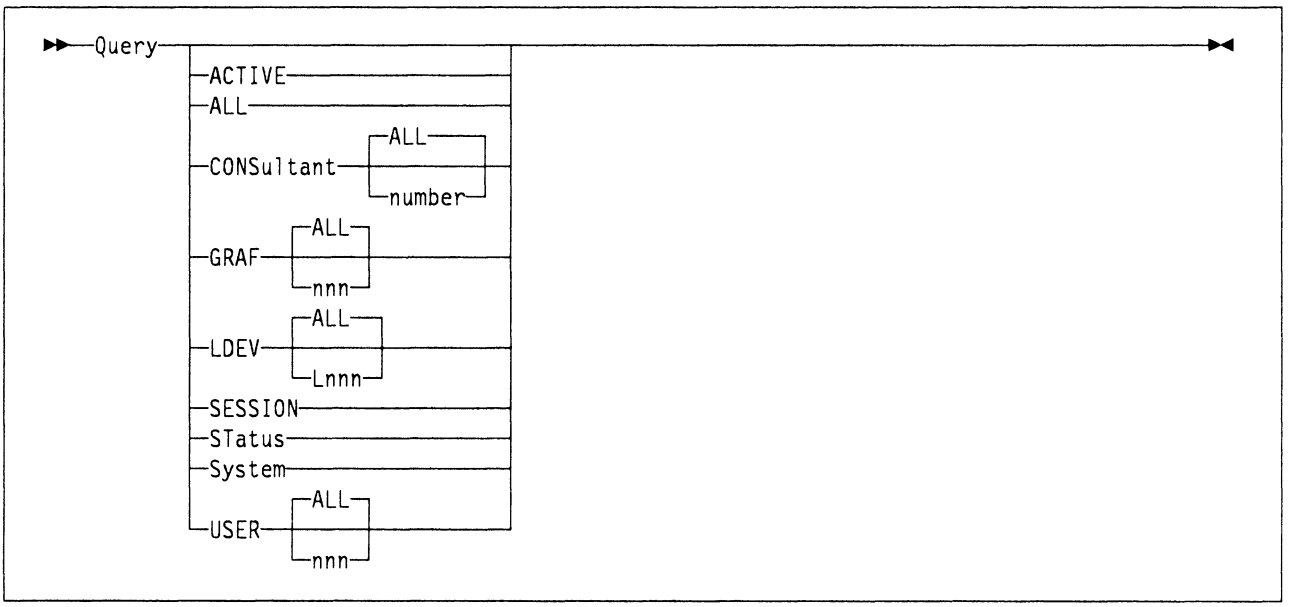
number
Is a value form 2 to 12 that corresponds to the consultant in the CONSULT IDNAME file.

newpassword
Is the new password to be assigned to the specified consultant. The new password can be up to eight characters long.

DISPLAY
Lets a privileged user look at a consultant's password.

number
Is a value from 2 to 12 that corresponds to the consultant in the CONSULT IDNAME file.

QUERY



Purpose

Gets information and status about many components of the CVIEW virtual machine.

Authority

General

Operands

Query
Used alone it returns the version, release, and modification level of the CVIEW virtual machine, if the CVIEW virtual machine is running.

ACTIVE
Returns the session name or the consultant's ID number, the session LDEV address, the display station type and model as assigned by CVIEW, and the display input PF key setting for all currently active consult and conference sessions. The QUERY ACTIVE command is a synonym for and interchangeable with the QUERY SESSION command.

ALL
Returns the same information as the QUERY ACTIVE command plus all the active GRAF addresses and the display station type and model of each GRAF. Also, ALL returns all customization values in effect.

The System option is a synonym for and is interchangeable with the ALL option.

CONSultant
Returns information about consultants.

number
Returns a busy or not busy status for a specific consultant listed in the CONSULT IDNAME file. It also states how many recipients are waiting for the consultant's help. Number 2 through 12 is the ID of a consultant in the ID NAME file.

ALL
Returns the same information that CONSultant number displays, for every consultant listed in the ID NAME file by consultant number. ALL is the default parameter.

GRAF
Returns the display station type and model and the session type if the GRAF is active. If the specified GRAF address is not active, the command returns a message indicating this.

nnn
Is the virtual address, assigned by the CVIEW virtual machine, of a 3270 display station. Valid GRAF addresses are 501 through 530.

ALL
Returns the display station type and model and the session type for every GRAF that is active. ALL is the default parameter.

LDEV
Returns the session name and type, the display station type and model as assigned by CVIEW, and the setting of the display-input PF key.

L*nnn*
Is the LDEV address assigned by CP to a CVIEW session. Use the QUERY ACTIVE command to obtain the LDEV address.

ALL
Returns the session name and type for all active LDEVs, along with their display station types and models as assigned by CVIEW, and the settings of the display-input PF key. ALL is the default parameter.

SESSION
Returns the same information as the QUERY ACTIVE command.

STatus
Returns a message indicating whether the CVIEW virtual machine is running.

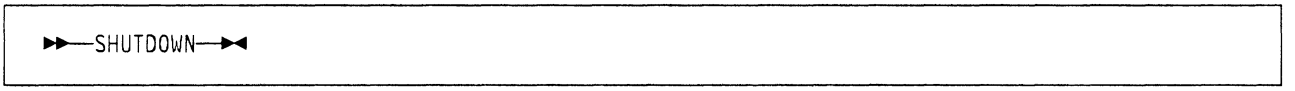
System
Returns the same information as the QUERY ALL command. The S option is a synonym for and interchangeable with the ALL option.

USER
Returns the same information as QUERY GRAF.

nnn
Is the virtual address, assigned by the CVIEW virtual machine, of a 3270 display station. Valid GRAF addresses are 501 through 530.

ALL
Returns the same information as the QUERY GRAF ALL command. ALL is the default parameter.

SHUTDOWN



Purpose

Stops the CVIEW virtual machine if no GRAFs or sessions are active. The CVIEW virtual machine will not stop if there are any active GRAFs or sessions.

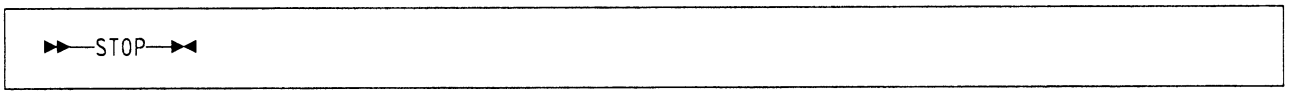
Authority

Privileged

Usage Notes

1. Use the SHUTDOWN command when you want to avoid affecting any active users and sessions. (See the STOP command.)

STOP



Purpose

Stops the CVIEW virtual machine. All active sessions and users are immediately ended.

Authority

Privileged

Usage Notes

1. When you enter a STOP command, no information is logged in the CVIEW LOG file. (See the SHUTDOWN command).

Establishing a Session

When the CVIEW operator has started the CVIEW virtual machine, consult and conference sessions can be established. As a session participant, you **do not** log on the CVIEW user-ID. Instead, you enter the 'DIAL CVIEW' command to set up and establish a session.

CVIEW provides menu and data-entry panels to guide you through the setup process. There are help panels for each setup panel; to see one press the PF key assigned to the HELP function. You will see several menu and data-entry panels in the course of setting up a CVIEW session.

When the consult or conference session starts, the logo for the VM/ESA system CVIEW is running under is displayed to all session participants. At this point, one participant with an active keyboard can clear the system logo and log on any user-ID defined in the system's directory or can "pass through" to another system node if desired.

To establish a consult or conference session, do the following:

1. Clear the VM/ESA system logo.
2. Enter: DIAL CVIEW

   The CVIEW HOME panel, shown in Figure 30 on page 131 appears.

```
HOME              COOPERATIVE VIEWING FACILITY (CVIEW)
------------------------------------------------------------------------------
 Enter an item number.

     Consult Session

         1  Receive assistance
         2  Give assistance

     Conference Session

         3  Attend a conference
         4  Moderate a conference




 Press ENTER after you enter an item number.
------------------------------------------------------------------------------
PF:  1= Help     2=          3= Quit     4=          5=          6=
PF:  7=          8=          9=         10=         11=         12=

    ===>
```

Figure 30. The HOME Panel Indicates the Start of Setup Mode

3. Select your role for the session from the HOME panel. Each role has specific characteristics:

   Consult Session

   - **Recipient's role — receive assistance**

     A recipient always has an active keyboard and can enter data and commands during the consult session. If the consultant assigns a display-input PF key, the recipient can use it. If the recipient presses the PA1 key, the host or application program processes it.

   - **Consultant's role — give assistance**

     A consultant always has an active keyboard and can enter data and commands during the consult session. A consultant can assign and use a display-input PF key. When the consultant presses the PA1 key, the CONSULT SESSION CONTROL panel appears and the consult session is suspended.

   Conference Session

   - **Attendee's role — attend a conference**

     An attendee always has an *inactive keyboard,* so he cannot enter data and commands during the conference session. He can, however, see the data and commands entered by the moderator and presenters, and can talk to other participants by telephone. If the moderator assigns him presenter status, he then has an active keyboard and can enter data and commands.

   - **Moderator's role — moderate a conference**

     A moderator always has an active keyboard and can enter data and commands during the conference session. A moderator can assign and use a display-input PF key. When the moderator presses the PA1 key, the CONFERENCE CONTROL panel appears and the conference session

is suspended. The moderator controls the session and can assign presenter status to any of the attendees.

See the discussions of consult and conference sessions, later in this chapter, for detailed descriptions of the activities associated with each role.

4. After you select a role from the HOME panel, additional setup panels are displayed to identify the consult or conference session. See Appendix K, "CVIEW Product Panels" on page 457, for all the setup-mode panels.
5. When setup is complete, the session starts: the logo of the VM/ESA system CVIEW is running under (similar to the logo in Figure 31 is displayed to all participants).

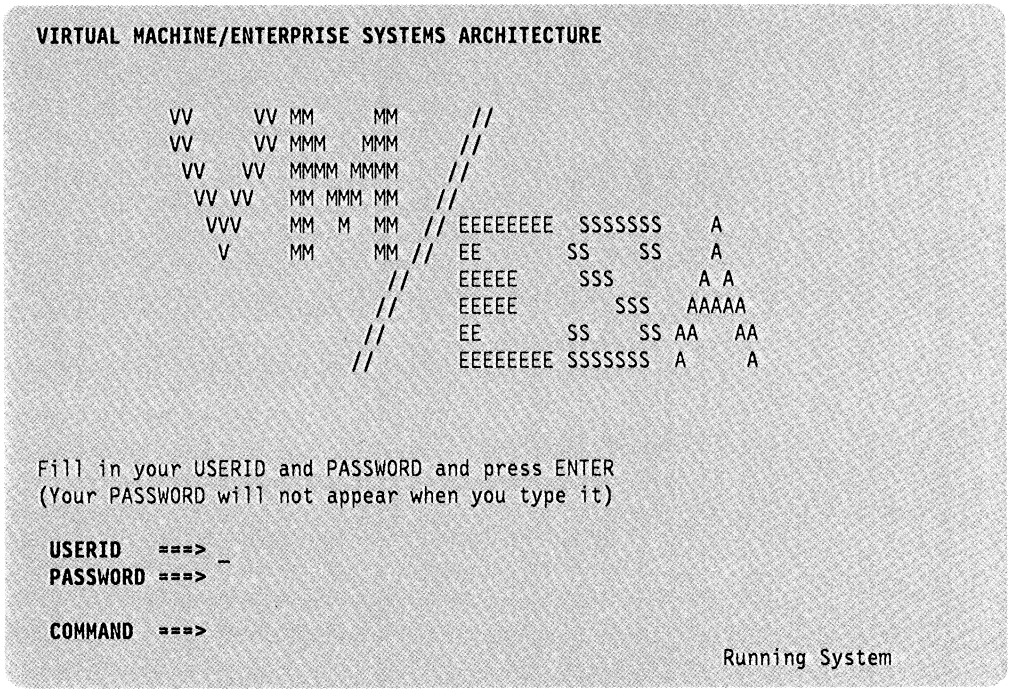


Figure 31. The VM/ESA System Logo. Indicates that the CVIEW session has started.

6. One session participant with an active keyboard clears the VM/ESA system logo logs on to the system, and proceeds as for a usual interactive session.

   For a consult session, either the recipient or the consultant can log on.

   For a conference session, the moderator or a presenter can log on.
7. If the notify option is in effect for your system, the NOTIFY panel appears for the first time shortly after one user logs on to the system. It will appear again at preset intervals as the interactive session progresses.

   Figure 32 shows the NOTIFY panel, which indicates the beginning of notify mode.

```
NOTIFY                          CVIEW - NOTIFY
------------------------------------------------------------------------------
 This Notify panel is being displayed by the Cooperative Viewing Facility
 (CVIEW) to make you aware you are running under the control of
 CVIEW.

 You are currently in a CVIEW session, someone else is watching the same
 screen images you are displaying.

 If you press the QUIT or HOME PF key while this panel is displayed, the
 session will be terminated.

 If you were not aware you were using this facility, contact your Computer
 Center Management.

 Every participant who receives a Notify panel must press ENTER for the
 session to continue

 Press ENTER to continue.
------------------------------------------------------------------------------
PF:  1= Help     2=          3= Quit     4= Home     5=          6=
PF:  7=          8=          9=         10=         11=         12=

    ===>
```

Figure 32. The NOTIFY Panel

If a NOTIFY panel appears on your screen, you must press the ENTER key for the session to continue. If you were not aware that your interactive session is executing under the control of CVIEW, and you do not wish to continue under these circumstances, press the PF key assigned to either the HOME or the QUIT function and the entire CVIEW session will end.

8. When the interactive session is finished, one user with an active keyboard logs off the system and the HOME panel is displayed to all session participants. Another session can be established or you can press the PF key assigned to the QUIT function to be dropped out of setup mode.

Managing a CVIEW Session

During setup and session modes, consult and conference sessions are controlled by the consultant and moderator. The consultant and moderator provide information to session participants during setup to correctly identify the session. In addition, the consultant and moderator can suspend a session and enter control mode to change certain session characteristics. The next few pages describe the things each user can do to manage a CVIEW session.

Consult Sessions

During setup mode, the consultant guides the recipient through the setup panels to provide the proper session identification. Consult-session identification can be either a listed consultant ID number or a name (if the consultant is not listed on the GIVE and RECEIVE panels). A listed consultant may also be required to enter a password. If the consultant is not listed, he must tell the recipient the exact session name to enter. The consult session starts when both users complete all the setup panels displayed to them.

A **recipient** can do the following:

- Return to the HOME panel during setup mode by pressing the HOME PF key
- Leave CVIEW during setup mode by pressing the QUITPF key
- Enter data or commands during the consult session
- Use the display-input PF key assigned by the consultant
- Send the PA1 key function to the host or an application program for processing, by pressing the PA1 key
- Continue a consult session by pressing ENTER in response to the NOTIFY panel
- End a consult session by pressing the QUIT or HOME PF key in response to the NOTIFY panel
- End a consult session by logging off.

A **consultant** can do the following:

- Return to the HOME panel during setup mode by pressing the HOME PF key
- Leave CVIEW during setup mode by pressing the QUIT PF key panel
- Enter data or commands during the consult session
- Assign a display-input PF key during setup mode
- Use the display-input PF key during the consult session
- Press the PA1 key to suspend the consult session and access the CONSULT SESSION CONTROL panel to:
  - Change or cancel the display-input PF key assignment
  - Send the PA1 key function to the host or an application program for processing, by pressing the PA1 key
  - End the consult session by pressing the HOME or QUIT PF key.
- Continue a consult session by pressing ENTER in response to the NOTIFY panel
- End the consult session by pressing the QUIT or HOME PF key in response to the NOTIFY panel
- End the consult session by logging off.

Conference Sessions

During setup mode for a conference session, the moderator guides attendees through the setup panels to provide the proper session identification. The moderator selects a session name and a password if one is desired. The moderator tells the attendees the exact session identification to enter. The moderator can assign presenter status to any of the attendees and drop attendees before or after starting the conference session. The moderator starts the conference by pressing the SESSION PF key when the CONFERENCE PARTICIPANTS panel is displayed.

A conference session **attendee**, whose keyboard is inactive, can do the following:

- Return to the HOME panel during setup mode by pressing the HOME PF key
- Leave CVIEW during setup mode by pressing the QUIT PF key

- Leave the conference session by pressing PA1. This does not disrupt the conference in progress if at least two participants remain.
- Continue a conference session by pressing ENTER in response to the NOTIFY panel
- End a conference session by pressing the QUIT or HOME PF key in response to the NOTIFY panel
- Join a conference already in progress except for a user with an incompatible display station type and model cannot join a conference session already in progress. See "Joining a Conference Session in Progress" on page 136

A conference session **presenter** can do the following:

- Return to the HOME panel during setup mode by pressing the HOME PF key
- Leave CVIEW during setup mode by pressing the QUIT PF key
- Enter data of commands during the conference session
- Use the display-input PF key assigned by the moderator
- Send the PA1 key function to the host or an application program for processing, by pressing the PA1 key
- Continue a conference session by pressing ENTER in response to the NOTIFY panel
- End a conference session by pressing the QUIT or HOME PF key in response to the NOTIFY panel
- End a conference session by logging off.

A conference session **moderator** can do the following:

- Return to the HOME panel during setup mode by pressing the HOME PF key
- Leave CVIEW during setup mode by pressing the QUIT PF key
- Enter data or commands during the conference session
- Assign a display-input PF key during setup mode
- Use the CONFERENCE PARTICIPANTS panel to:
  - Start a conference session by pressing the SESSION PF key during setup mode
  - Change an attendee's status to presenter
  - Change a presenter's status to attendee
  - Drop an attendee who is not on the NOTIFY panel
  - Return to a conference session by pressing the SESSION PF key
  - End the session by pressing the HOME or QUIT PF key
  - End the session by dropping the last attendee.
- Use the display-input PF key during the conference session
- Press the PA1 key to suspend the conference session and access the CONFERENCE CONTROL panel. From the CONFERENCE CONTROL panel the moderator can:
  - Change or cancel the display-input PF key assignment
  - Send the PA1 key function to the host or an application program for processing

- End the session by pressing the HOME or QUIT PF key
- Display the CONFERENCE PARTICIPANTS panel.

- Continue a conference session by pressing ENTER in response to the NOTIFY panel
- End a conference session by pressing the QUIT or HOME PF keys in response to the NOTIFY panel
- End a conference session by logging off.

Device Type and Model Assignment

When a consult or conference session is being established, CVIEW determines the display station type and model to be used in the session. Its purpose is to provide all session participants with identical screen images while in session mode. Because a variety of display stations with different screen sizes are supported by CVIEW, it must assign a display station type and model to the session that accommodates the different display stations participating in the session. The display station type and model assigned to the session is determined as session mode starts.

The following table shows the display station type and model selected for a session when two types and models are compared.

	3277-2	**3278-2**	**3278-3**	**3278-4**	**3278-5**
3277-2	3277-2	3277-2	3277-2	3277-2	3277-2
3278-2	3277-2	3278-2	3278-2	3278-2	3277-2
3278-3	3277-2	3278-2	3278-3	3278-3	3277-2
3278-4	3277-2	3278-2	3278-3	3278-4	3277-2
3278-5	3277-2	3277-2	3277-2	3277-2	3278-5

For example, if a consultant has a 3278-4 and the recipient has a 3278-2, CVIEW assigns a display station type and model of 3278-2 to the consult session.

Joining a Conference Session in Progress

The following table illustrates whether a display station wanting to join a conference session will be allowed to join if the moderator has started the session.

The horizontal header represents the display station type and model assigned to the conference session.

The leftmost column represents the type and model of a display station that wants to join a conference after the session has begun.

	3277-2	**3278-2**	**3278-3**	**3278-4**	**3278-5**
3277-2	Yes	No	No	No	No
3278-2	Yes	Yes	No	No	No
3278-3	Yes	Yes	Yes	No	No
3278-4	Yes	Yes	Yes	Yes	No
3278-5	Yes	No	No	No	Yes

For example, a conference session has been started by the moderator and CVIEW has assigned the display station type and model as a 3278-3. A user wants to join the conference in progress from a 3278-5. The user will not be allowed to join the conference in progress from that display station. The user could join the conference in progress from a 3278-3 or a 3278-4.

The 3278-5 has a wider screen than the other display station types and models supported by CVIEW Version 2. If the entire 3278-5 screen width is needed in a session, it can be used only if all display stations participating in the session are 3278-5.

CVIEW PROFILE EXEC

CVIEW includes a sample profile exec, named DXGPROF EXEC. The exec is loaded to the CVIEW 191 A-disk during installation. The control statements contained in the sample profile exec are as follows:

```
&TRACE OFF
* 5684-100 COPYRIGHT IBM CORP 1993
* LICENSED MATERIAL - PROGRAM PROPERTY OF IBM
EXEC DXGCVIEW
&EXIT &RETCODE
```

To use the sample profile exec, copy the file with the replace option and specify the new filename and filetype as PROFILE EXEC. **Do not** erase or rename the DXGPROF EXEC file on the CVIEW 191 A-disk.

CVIEW Log File

CVIEW includes a logging file that records information about consult and conference session activities. The CVIEW virtual machine writes records to the CVIEW LOG file sequentially as activity occurs, until the CVIEW A-disk is full. When the A disk is full, no more information is logged in the CVIEW LOG file. The person responsible for administering the CVIEW virtual machine should periodically review and purge the file.

You can use the information in the CVIEW LOG file to generate utilization reports for your establishment. The file contains variable-length records. The fields within each record contain the following:

Positions	Field Content
1 - 5	Type of session activity
7 - 16	Type of session
18 - 25	Session identification
27 - 36	User's role in session
38 - 45	Date activity occurred
47 - 54	Time activity occurred
56 - 61	Display-station model number as assigned by CVIEW for GRAF or session

Positions	Field Content
63 - end	One of the following: • CIVIEW virtual GRAF address and the real address • CVIEW virtual GRAF address and the logical device address • CVIEW virtual GRAF addresses for the GRAF devices in a conference session.

The following are sample records from a CVIEW LOG file:

```
Record <....+....1....+....2....+....3....+....4....+....5....
Number +....6....+....7....+....8....+....9....+....10...+....>

00001   JOIN  CONFERENCE SALES    Moderator  02/13/93 15:48:32
        3277-2 501 = LDEV E06

00002   JOIN  CONFERENCE SALES    Attendee   02/13/93 15:50:34
        3278-2 502 = GRAF 391

00003   JOIN  CONFERENCE SALES    Attendee   02/13/93 15:50:45
        3278-3 503 = GRAF 392

00004   START CONFERENCE SALES    Session    02/13/93 16:35:18
        3277-2 GRAFS=  501  502  503

00005   STOP  CONFERENCE SALES    Session    02/13/93 16:35:39
        3277-2 GRAFS=  501  502  503

00006   JOIN  CONSULT    2        Recipient  03/07/93 10:56:17
        3278-2 501 = GRAF 383

00007   JOIN  CONSULT    2        Consultant 03/07/93 10:56:26
        3278-3 502 = GRAF 344

00008   JOIN  CONSULT    2        Session    03/07/93 10:56:27
        3278-2 GRAFS=  502  501

00009   STOP  CONSULT    2        Session    03/07/93 10:58:31
        3278-2 GRAFS=  502  501

00010   JOIN  CONFERENCE HISTORY  Attendee   03/07/93 10:59:00
        3278-4 501 = GRAF 383

00011   JOIN  CONFERENCE HISTORY  Moderator  03/07/93 10:59:06
        3278-5 502 = GRAF 344

00012   START CONFERENCE HISTORY  Session    03/07/93 10:59:26
        3277-2 GRAFS=  502  501

00013   STOP  CONFERENCE HISTORY  Session    03/07/93 11:04:55
        3277-2 GRAFS=  502  501
```

Part 2. Servicing and Determining Problems

Chapter 11. Applying Service to PVM 2.1 141
Applying Service Using VMSES/E 141
Applying Service without VMSES/E 141
Applying Preventative Service (PUT Tape) 142
Applying Corrective Service 147
Preparing To Run CORRSERV 172
Understanding How CORRSERV Processes 173
CORRSERV 175

Chapter 12. Collecting Problem Information 179
Trace Facility for the PVM Component 180
The Dump Facility 187
Analyzing Dump Information 189
Diagnosing PVMG Problems 193

© Copyright IBM Corp. 1991,1993

Chapter 11. Applying Service to PVM 2.1

This chapter provides you with information to help you apply both preventative and corrective service to your PVM installation either with or without VMSES/E.

The IBM Support Center delivers (upon request) solutions to PVM problems. When you find a problem with PVM, MPVM, or PVMG, contact the IBM Support Center with a description of the problem. Chapter 12, "Collecting Problem Information" helps you gather information about a problem and report that problem to the Support Center.

Updating National Language Support Files

To update a message repository file for the PVM or PVMG modules, enter:

```
pvmcmp dvm langid ctl pvm
```

If the updates are for the EFGVIEW (MPVM) or DVMUSI (PASSTHRU) modules, enter:

```
pvmcmp efg langid ctl pvm
```

langid is the language ID of the message file to be updated.

Also, if there are any updates for the PVMDEST FILE, all message repository files for PVM and PVMG have to be recompiled using the PVMCMP command. These only have to be recompiled for the repository that is used for the default language.

Applying Service Using VMSES/E

If you have PVM 2.1.1 PVM is serviced (and installed) using the Virtual Machine Serviceability Enhancements Staged/Extended (VMSES/E) component. It provides you with the ability to:

- Verify prerequisites for product installation
- Install PVM, with or without replacing an earlier release
- Migrate to a new release, with or without replacing an earlier release
- Apply service to a product
- Delete a product

For complete details on VMSES/E, see *VM/ESA: VMSES/E Introduction and Reference*.

To use VMSES/E see your PVM program directory for instructions. It may be a good idea to save the program directory if you need to refer to it later.

Applying Service without VMSES/E

There are two kinds of service available from the Support Center: preventative and corrective.

1. Preventative service consists of accumulating changes made to PVM and updating the product to its highest modification level. These fixes and updates are placed on a Program Update Tape (PUT). PUTs help prevent problems from recurring.

© Copyright IBM Corp. 1991,1993

2. Corrective service consists of Program Temporary Fixes (PTFs). PTFs are solutions to PVM problems that are made available before they are placed on a PUT.

Apply programming service after PVM has been installed. Use the MAINT user ID to load the service files onto your system.

Applying Preventative Service (PUT Tape)

Before you install the service tape, make sure that you have defined and formatted a Service Staging disk (address 29E) for the MAINT user ID.

If a PVM minidisk was not loaded during the initial installation of PVM Version 2, that minidisk should be passed when prompted for loading by the service exec. The service tape contains only files that have changed (that is not a total inclusive set of files).

All CMS minidisks for PVM Version 2 must be CMS formatted. This was completed at the time of initial installation. The only minidisk that may be unformatted is the PVM service disk (default disk is 29E). If the service exec finds this disk unformatted, the exec will format the disk before loading the service files. If you are applying service to PVM Version 2 for the first time, and if you are loading the service files from tape, you will have to define a PVM service minidisk in MAINT's CP directory entry.

The service exec looks for a version-specific identification string in a PVM module that is found on the PVM system minidisk. If it does not find this string, the service exec will ask if you want to continue applying service.

Although there are 13 files on the service tape, some files may be dummy files to ensure a consistent tape format. Dummy files indicate that there is no service necessary for those files.

If you wish to apply local modifications such as PVM user exits, you must load tape files 3 through 7, using the 5684100A service exec. Then reassemble any PVM source modules that were *TOUCH*ed during the modification. Finally, build the new PVM, PVMG, DVMUSI, and EFGVIEW modules using the PVMBLD , PVMGBLD, PVMGEN, and MPVMBLD commands, respectively.

The PUT includes a service exec. For the base PVM, this exec is called 5684100A EXEC. This service exec helps you to load the base PVM service from the PVM service tape. Each PUT tape also includes a Memo-to-Users document that contains instructions for applying the service updates.

These are ways to apply service for PVM:

- VMSERV EXEC, which calls the base service exec
- 5684100A EXEC directly loaded
- Enter the low-level tape commands explicitly.

Most PUT users will want to use the VMSERV EXEC for application of the PUT tape. If your installation does not have the VMSERV EXEC, you will have to load service file from the PUT using tape commands.

The format of the service tape is in Appendix B, "Tape Layouts for PVM 2.1." The mandatory files on the PUT are for the system extension minidisk and the PVM system minidisk.

Applying Preventative Service Using VMSERV

1. Log on MAINT. If you are not already logged on as MAINT enter the following commands:

   ```
   logon maint
   (Enter password)
   ipl cms
   ```

2. Mount the PVM service (PUT) tape and attach the tape drive to MAINT as 181. Issue the following commands:

   ```
   CP ATTACH raddr MAINT 181
   CP REWIND 181
   ```

   Substitute the tape drive address for *raddr*.

3. Access the MAINT's 191 disk as C. Enter the following command:

   ```
   access 191 c
   ```

4. Start the VMSERV EXEC. Enter the following command:

   ```
   vmserv
   ```

   VMSERV prints the memo-to-users and exits.

5. Read the PVM memo-to-users.

6. Start the VMSERV EXEC again. Enter the following command:

   ```
   vmserv
   ```

   VMSERV asks you if you want to service PVM. When you answer YES, VMSERV starts the service exec (5684100A EXEC or 5684NLSA EXEC) and the service exec finishes loading the PVM service tape.

   The service exec displays more questions at your display station as it runs. Answer these questions as appropriate for your installation. As the service exec processes, you will be prompted for minidisk information with prompts similar to:

   ```
   -36E- | ADDR | LINK | QUIT | PASS
   ```

   where:

   -36E-
   allows the user to select the default MAINT 36E minidisk as the load minidisk address by pressing ENTER

   ADDR
   allows the user to specify a minidisk address other than the default. This minidisk is assumed to be formatted.

   LINK
   allows the user to dynamically link to the necessary minidisk The exec will prompt the user for *MR_PASSWORD*, *USERID's* virtual minidisk *ADDRESS*.

   QUIT
   used to stop the service exec.

PASS
lets the user bypass loading the current tape file. This is not accepted for the PVM system disk and the PVM system extension disk since these files must be loaded for service.

Note: The service exec will rename the PVM MODULE, DVMUSI MODULE, and the EFGVIEW MODULE (if installed) to a filename of OPVM, ODVMUSI, or OEFGVIEW. The PVMG module is not renamed.

After applying service with 5684100A EXEC, there is no need to rebuild the new PVM, EFGVIEW, PVMG, and DVMUSI modules. These modules are loaded as part of the servicing process.

Running the Service Exec Without VMSERV

You may run the service exec without using VMSERV by:

- Accessing 191 as C
- Mounting the service tape as 181
- Issuing

```
REW 181
VMFPLC2 SCAN 5684100A MEMO (EOT
VMFPLC2 LOAD * * C
```

- and finally, issuing

```
5684100A
```

The 5684100A service exec will prompt you for the necessary information as described in the preceding section.

Applying Preventative Service Standalone

Use the following procedure to load base PVM service:

1. Log on MAINT. If you are not already logged on as MAINT enter the following commands:

```
logon maint
(Enter password)
ipl cms
```

2. Mount the PVM service (PUT) tape and attach the tape drive to MAINT as 181. Enter the following commands:

```
CP ATTACH raddr MAINT 181
CP REWIND 181
```

Substitute the tape drive address for *raddr*.

3. Access the MAINT's 191 disk as C. Enter the following command:

```
access 191 c
```

4. Move the tape to the correct position for corrective service for PVM. Move to the start of the service exec. Enter the following commands:

```
vmfplc2 scan 5684100A memo (eot
vmfplc2 fsf 1
```

   Load tape file #2 to the 191 disk, print, and read the memo-to-users. To finish loading the PVM service tape standalone, enter:

   a. Load tape file #3 onto the MAINT 191 disk:

```
ACCESS 191 A
VMFPLC2 LOAD * * A
```

      This tape file must be loaded. 191 is the address of your sample directory.

   Access the following disks:

   - MAINT 191
   - MAINT 19D
   - MAINT 19E
   - MAINT 36E
   - MAINT 38F
   - MAINT 39E
   - MAINT 38E
   - MAINT 49E
   - MAINT 29E
   - MAINT 298.

   You will be prompted for the MR passwords when linking to these disks later.

5. Place tape file #4 (PVM system files) on MAINT 36E:

```
ACCESS 36E A
VMFPLC2 LOAD * * A
```

6. Place tape file #5 (System Extension files) on MAINT 19E:

```
ACCESS 19E A
VMFPLC2 LOAD * * A
```

7. Place tape file #6 (PVM source files) on MAINT 39E:

```
ACCESS 39E A
VMFPLC2 LOAD * * A
```

   Or, if you wish to bypass loading of these files:

```
VMFPLC2 FSF 1
```

8. Place tape file #7 (PVM Service Files) on MAINT 29E:

```
ACCESS 29E A
VMFPLC2 LOAD * * A
```

   Or, if you wish to bypass loading of these files:

```
VMFPLC2 FSF 1
```

9. Place tape file #8 (PVM base text files) on MAINT 49E:

   ```
   ACCESS 49E A
   VMFPLC2 LOAD * * A
   ```

   Or, if you wish to bypass loading of these files:

   ```
   VMFPLC2 FSF 1
   ```

10. Place tape file #9 (PVM help files) on MAINT 19D:

    ```
    ACCESS 19D A
    VMFPLC2 LOAD * * A
    ```

    Or, if you wish to bypass loading of these files:

    ```
    VMFPLC2 FSF 1
    ```

11. Place tape file #10 (IPCS help and text files) on MAINT 193:

    ```
    ACCESS 193 A
    VMFPLC2 LOAD * * A
    ```

    Or, if you wish to bypass loading of these files:

    ```
    VMFPLC2 FSF 1
    ```

12. Tape file #11 contains PVMG files. Use the following commands to place files on MAINT's 38E:

    ```
    ACCESS 38E A
    VMFPLC2 LOAD * * A
    ```

    If you *did not* install PVMG, enter the following command to bypass loading these files:

    ```
    VMFPLC2 FSF 1
    ```

13. Tape file #12 contains the MPVM System files. Use the following commands to place files on MAINT's 38F disk:

    Place tape file #12 on the MAINT 38F disk by issuing:

    ```
    ACCESS 38F A
    VMFPLC2 LOAD * * A
    ```

    Or, if you did *not* install MPVM, enter this command to bypass loading these files:

    ```
    VMFPLC2 FSF 1
    ```

14. Place tape file #13 on the MAINT 298 disk (VTAM files for PVMG) by issuing:

    ```
    ACCESS 298 A
    VMFPLC2 LOAD * * A
    ```

    (VTAM links to the MAINT 298 disk as its 191 disk.)

15. Rewind and detach the tape:

    ```
    CP REWIND 181
    CP DETACH 181
    ```

16. Get MAINT's A-disk back:

    ```
    ACCESS 191 A
    ```

Procedures for loading the base PVM service is also documented in the Memo-to-Users, which is provided with each PUT service tape. These descriptions may reflect additional changes in the service install procedure.

Applying Corrective Service

A *corrective* service tape contains service that:

- Results from closed valid APARs. This is on a Program Temporary Fix (PTF) tape or has been received electronically and is supplied by Program Support Services (PSS).
- Is a *relief fix* that is on a tape provided by an IBM Change Team prior to closing the corresponding APAR (or APARs). This may not be in correct format. Ask your Change Team representative for instructions.

To manually apply service, the procedure contains three parts for:

1. For Module and Loadlib replacement users,
2. For Text replacement users, or
3. For Source replacement users.

Note: See "Location of PVM's Files on MAINT's Minidisks" on page 175 for the location of PVM files on Maint's minidisks.

Manually Applying Corrective Service for MODULE and LOADLIB Users

To complete this job successfully, observe all instructions and commands included in each step. Commands should be entered as shown except where *variable* information is requested.

Step 1. Call your IBM Support Center for the latest service information about the type of service you are installing.

Step 2. Read the letter supplied with the service tape for additional information about this specific tape. You can find a machine readable copy of the letter in Tape File 1 of the service tape.

Step 3. Log on to the MAINT virtual machine using the following commands:

```
logon maint
Enter MAINT's logon password.
ipl cms
```

Step 4. Verify that you have the correct minidisks available:

36E	PVM System minidisk
49E	PVM Text minidisk
38E	PVMG System minidisk
38F	MPVM System minidisk
19E	System Extension Files minidisk
193	IPCS Text minidisk
19D	PVM VM/ESA Help minidisk (optional)

Step 5. Access the 191 minidisk as C enter the following command:

```
acc 191 c
```

Step 6. Create a temporary work minidisk and access it as file mode A using the following commands:

```
cp define Tttttt ccuu nnn
format ccuu a
```

Where *tttt* is the temporary work minidisk's device type, *ccuu* is the address of the work minidisk, and *nnn* is the size of the work minidisk in cylinders or blocks. (**Note:** The size of the work minidisk you define will

depend on the amount of service you ordered. If you ordered a lot of service, you should define a large work minidisk.)

Step 7. Mount and attach the PVM corrective service tape to MAINT as 181.

```
cp attach ccuu to maint as 181
cp rewind 181
```

Where *ccuu* is the tape drive real address.

Step 8. Position the tape to the PVM product files using the following command:

```
vmfplc2 scan 5684100A 010101 (eot
```

Step 9. Load the PVM service execs and related files to the temporary work minidisk using the following command:

```
vmfplc2 load * * a
```

This command loads all the files on the tape to your temporary work minidisk.

Step 10. Look for an updated service exec ("5684100A EXC*") on the temporary work minidisk using the following command:

```
listfile 5684100A exc* a
```

If the file is not found, continue with the next step. If the file is found, check to see if you have an existing service exec on your file mode C minidisk (191 minidisk) using the following command:

```
listfile 5684100A exec c
```

If you find an existing service exec, rename it using the following command:

```
rename 5684100A exec c = oexec =
```

Notes:

1. Initially, there will be no 5684100A EXEC file on the 191 minidisk.
2. If you have already applied service to PVM, an OEXEC file may already exist on the 191 minidisk. If this file exists, you can erase it (or save it on a tape).

If you do not find an existing service exec or if you have already renamed it, copy the updated service exec to your file mode C minidisk (191 minidisk) and change the file type to EXEC:

```
copy 5684100A excnnnnn a = exec c (olddate
```

Then erase it from the temporary work minidisk.

Step 11. Invoke the rename exec supplied on the tape using the following command:

```
5684100A a
```

This exec renames the replacement parts from the tape to the correct file type and erases any service elements shipped on the tape that were not renamed.

Step 12. Access the 36E minidisk as file mode B using the following command:

```
acc 36e b
```

Note: Skip the next two steps if a new PVM MODULE does not exist on the service tape.

Step 13. Rename the existing PVM MODULE, MAP, and PRELMAP files on the 36E minidisk using the following commands:

```
rename pvm module b pvm omodule =
rename pvm map b pvm omap =
rename pvm prelmap b pvm oprelmap =
```

Note: If you have already applied service to PVM, OMODULE, OMAP, and OPRELMAP files may already exist on the 36E minidisk. If these files exist, you can erase them (or save them on a tape).

Step 14. Copy the PVM MODULE, MAP, and PRELMAP files from the temporary work minidisk to the 36E minidisk using the following commands:

```
copy pvm module a = = b (olddate
copy pvm map a = = b (olddate
copy pvm prelmap a = = b (olddate
```

Step 15. Access the 38F minidisk as file mode B using the following command:

```
acc 38f b
```

Note: Skip the next two steps if new EFGVIEW and EFGVSTUB MODULEs do not exist on the service tape.

Step 16. Rename the existing EFGVIEW MODULE, MAP, PRELMAP, TEXT, and EFGVSTUB MODULE files on the 38F minidisk using the following commands:

```
rename efgview module b efgview omodule =
rename efgvstub module b efgvstub omodule =
rename efgview text b efgview otext =
rename efgview map b efgview omap =
rename efgview prelmap b efgview oprelmap =
```

Notes:

1. You should only rename those files for which there are new replacements on the service tape.
2. If you have already applied service to EFGVIEW or EFGVSTUB, OMODULE, OMAP, OPRELMAP, and OTEXT, files may already exist on the 38F minidisk. If these files exist, you can erase them (or save them on a tape).

Step 17. Copy the EFGVIEW MODULE, MAP, PRELMAP, TEXT, and EFGVSTUB MODULE files from the temporary work minidisk to the 38F minidisk using the following commands:

```
copy efgview module a = = b (olddate
copy efgview text a = = b (olddate
copy efgvstub module a = = b (olddate
copy efgview map a = = b (olddate
copy efgview prelmap a = = b (olddate
```

Step 18. Access the 38E minidisk as file mode B using the following command:

```
acc 38e b
```

Note: Skip the next two steps if a new PVMG LOADLIB does not exist on the service tape.

Step 19. Rename the existing PVMG LOADLIB and LKEDIT files on the 38E minidisk using the following commands:

```
rename pvmg loadlib b pvmg oloadlib =
rename pvmg lkedit b pvmg olkedit =
```

Notes:

1. Initially, there will be no PVMG LKEDIT file on the 38E minidisk.
2. If you have already applied service to PVMG, OLOADLIB and OLKEDIT files may already exist on the 38E minidisk. If these files exist, you can erase them (or save them on a tape).

Step 20. Copy the PVMG LOADLIB and LKEDIT files from the temporary work minidisk to the 38E minidisk using the following commands:

```
copy pvmg loadlib a = = b (olddate
copy pvmg lkedit a = = b (olddate
```

Note: Initially, there will be no PVMG LKEDIT file on the 38E minidisk.

Step 21. Access the 19E minidisk as file mode B using the following command:

```
acc 19e b
```

Note: Skip the next two steps if the new DVMUSI MODULE does not exist on the service tape.

Step 22. Rename the existing DVMUSI file on the 19E minidisk using the following commands:

```
rename dvmusi module b dvmusi omodule =
rename dvmusi map b dvmusi omap =
rename dvmusi prelmap b dvmusi oprelmap =
```

Notes:

1. You should only rename those files for which there are new replacements on the service tape.
2. If you have already applied service to DVMUSI, OMODULE, OMAP, and OPRELMAP files may already exist on the 19E minidisk. If these files exist, you can erase them (or save them on a tape).

Step 23. Copy the DVMUSI file from the temporary work minidisk to the 19E minidisk using the following commands:

```
copy dvmusi module a = = b (olddate
copy dvmusi map a = = b (olddate
copy dvmusi prelmap a = = b (olddate
```

Step 24. Did you load the optional PVM VM/ESA Help files during PVM installation?

If you **did not** load the optional PVM VM/ESA Help files, you can skip the next three steps. If you **did** load the optional PVM VM/ESA Help files, continue with the following steps.

Step 25. Determine if there are any files on the service tape that affect the optional PVM VM/ESA Help minidisk (19D) using the following command:

```
listfile * help* a
```

If no files are found, you can skip the next two steps. If Help files are found, continue with the following steps.

Step 26. Access the optional PVM VM/ESA Help minidisk as file mode B using the following command:

```
acc 19d b
```

Step 27. Copy the affected files from the temporary work minidisk to the PVM VM/ESA Help minidisk using the following commands:

```
copy * helpabbr a = = b (olddate replace
copy * helppvm a = = b (olddate replace
copy * helpmenu a = = b (olddate replace
copy * helpmsg  a = = b (olddate replace
```

Step 28. Determine if there are any other files on the service tape that you need. There may be replacement files on the service tape to correct problems with files related to installation, service, execs, MPVM macros, or sample files. If, during installation, you loaded these files onto your disks, you will need to load the replacement files from the service tape to those same disks.

The following list shows the file types you may see on the service tape. (It is not a complete list. It is only a sample of some of the existing file types.)

$TABLE EXEC MPVM NAMESAMP
XEDIT

If you find some of these files on the service tape, continue with the following steps. If you do not find any of these files on the service tape, you have completed the manual application of corrective service for module and loadlib replacement users and can skip the remaining steps.

Step 29. Access all of the minidisks that you used in the previous steps using the following commands:

```
acc 36e b
acc 49e d
acc 38f g
acc 38e h
acc 19e i
acc 193 j
acc 19d k
```

Notes:

1. Do not use file mode A, because it is already being used by the temporary work minidisk.
2. Do not use file mode C, because it is already being used by the 191 minidisk.
3. The commands listed above show a specific order and a specific file mode letter. The order and the file mode letter are not important. You can access these disks using any file mode (except A or C) and in any order. Thus, if any of these file modes are already in use, you can substitute any other file mode letter that is not currently being used.

Step 30. List the files on the temporary work minidisk using the following command:

```
filelist * * a
```

After the command completes, it may be helpful to sort the list by file type using the PF4 key.

Step 31. Scan the files on the temporary work minidisk to find any that you have not already copied. You can ignore the files with file types that start with HELP, because you have already handled these files in the previous steps. Also, you can ignore the

- PVM MODULE, MAP, and PRELMAP
- DVMUSI MODULE, MAP, and PRELMAP
- EFGVIEW MODULE, MAP, PRELMAP, and TEXT
- EFGVSTUB MODULE
- PVMG LOADLIB, and LKEDIT
- the rename exec
- the servicing files

If you do find some of these extra files on the temporary work minidisk, continue with the following step. If you do not find any of these extra files on the temporary work minidisk, you have completed the manual application of corrective service for module and loadlib replacement users and can skip the remaining step.

Step 32. Compare the two files and find the newest one using the following command:

```
filelist fn ft *
```

If the date and time on the file on the temporary work minidisk is the newest version, copy it to the other minidisk using the following command:

```
copy fn ft a = = newfm (olddate replace
```

Where *newfm* is the file mode of the minidisk with the older version of the file.

This completes the manual application of corrective service for module and loadlib replacement users. For more information about servicing VM, including details of the VMFPLC2 command, see the *VM/ESA Service Guide*, and the *VM/ESA Service Guide for 370* books.

Manually Applying Corrective Service for TEXT Users

To complete this job successfully, observe all instructions and commands included in each step. Commands should be entered as shown except where *variable* information is requested.

Step 1. Call your IBM Support Center for the latest service information about the type of service you are installing.

Step 2. Read the letter supplied with the service tape for additional information about this specific tape. You can find a machine readable copy of the letter in Tape File 1 of the service tape.

Step 3. Log on to the MAINT virtual machine using the following commands:

```
logon maint
Enter MAINT's logon password.
ipl cms
```

Step 4. Verify that you have the correct minidisks available:

36E	PVM System minidisk
49E	PVM Text minidisk
39E	PVM Source minidisk
29E	PVM Service minidisk (optional)
38E	PVMG System minidisk
38F	MPVM System Files minidisk
19E	PVM System Extension files minidisk
193	IPCS Text minidisk
19D	PVM VM/ESA Help minidisk (optional)

Step 5. Access the 191 minidisk as C using the following command:

```
acc 191 c
```

Step 6. Create a temporary work minidisk and access it as file mode A using the following commands:

```
cp define tttt ccuu nnn
format ccuu a
```

Where *tttt* is the temporary work minidisk's device type, *ccuu* is the address of the work minidisk, and *nnn* is the size of the work minidisk in cylinders or blocks. (**Note:** The size of the work minidisk you define will depend on the amount of service you ordered. If you ordered a lot of service, you should define a large work minidisk.)

Step 7. Mount and attach the PVM corrective service tape to MAINT as 181.

```
cp attach ccuu to maint as 181
cp rewind 181
```

Where *ccuu* is the tape drive real address.

Step 8. Position the tape to the PVM product files using the following command:

```
vmfplc2 scan 5684100A 010101 (eot
```

Step 9. Load the PVM service execs and related files to the temporary work minidisk using the following command:

```
vmfplc2 load * * a
```

This command loads all the files on the tape to your temporary work minidisk.

Step 10. Look for an updated service exec ("5684100A EXC*") on the temporary work minidisk using the following command:

```
listfile 5684100A exc* a
```

If the file is not found, continue with the next step. If the file is found, check to see if you have an existing service exec on your file mode C minidisk (191 minidisk) using the following command:

```
listfile 5684100A exec c
```

If you find an existing service exec, rename it using the following command:

```
rename 5684100A exec c = oexec =
```

Notes:

1. Initially, there will be no 5684100A EXEC file on the 191 minidisk.
2. If you have already applied service to PVM, an OEXEC file may already exist on the 191 minidisk. If this file exists, you can erase it (or save it on a tape).

If you do not find an existing service exec or if you have already renamed it, copy the updated service exec to your file mode C minidisk (191 minidisk) and change the file type to EXEC:

```
copy 5684100A excnnnnn a = exec c (olddate
```

Then erase it from the temporary work minidisk.

Step 11. Invoke the rename exec supplied on the tape using the following command:

```
5684100A a
```

This exec renames the replacement parts from the tape to the correct file type and erases any service elements shipped on the tape that were not renamed.

Step 12. Access the 49E minidisk as file mode B, the 39E minidisk as file mode D, and the 19E minidisk as file mode E using the following command:

```
acc 49e b
acc 39e d
acc 19e e
```

Step 13. List the TEXT files and determine which are to be used when building the PVM, DVMUSI, EFGVIEW, and EFGVSTUB MODULEs, and the PVMG LOADLIB.

```
filelist * text *
filelist * txtsec *
```

If there are multiple copies (the same file exists on two different disks), you should manually check each file (using XEDIT) to determine which one is at the higher level (which file has more APARs listed at the top of the file).

If the higher level file resides on the temporary work minidisk, rename the existing one on the 49E minidisk for TEXT file types, and on the 39E minidisk for TXTSEC file types, and copy the file from the temporary work minidisk to the appropriate minidisk using the COPYFILE command with the OLDDATE option:

```
rename fn text b fn newft b
rename fn txtsec d fn newft d
copyfile fn text a = = b (olddate
copyfile fn txtsec a = = d (olddate
```

Where *fn* is the file name of the file and *newft* is the new file type of the file.

Notes:

1. The TXTSEC file types are for the Access Security Exits sample code.

2. The following TEXT files, associated with the IPCS facility, should be copied to the 193 minidisk instead of the 49E minidisk:

```
DVMZCO   TEXT
DVMZDS   TEXT
DVMZDV   TEXT
DVMZEX   TEXT
DVMZLI   TEXT
DVMZRO   TEXT
DVMZSY   TEXT
DVMZTK   TEXT
DVMZTR   TEXT
DVMZUS   TEXT
CSIYTXDC TEXT
```

Step 14. Copy any PVM, DVMUSIE, and EFGVIEW EXECs, and PVM, MPVM, and PVMG CNTRL files from the temporary work minidisk to the 49E minidisk, and the PVMSEC CNTRL and PVMG LKEDCTRL files to the 39E minidisk, using the following commands:

```
copy pvm exec a = = b (olddate replace
copy dvmusie exec a = = b (olddate replace
copy efgview exec a = = b (olddate replace
copy pvm cntrl a = = b (olddate replace
copy mpvm cntrl a = = b (olddate replace
copy pvmg cntrl a = = b (olddate replace
copy pvmsec cntrl a = = d (olddate replace
copy pvmg lkedctrl a = = d (olddate replace
```

Notes:

1. If your installation has previously installed one of the languages from the national language features tape, you must ensure that the proper languages are specified in the build execs. For more information, see "Step 3. Enabling PVM Components for Additional Languages" on page 233.
2. If your installation is using the Access Security Exits sample code, you must ensure that proper text files are specified in the build execs. For more information, see "Installing and Building the Security Exits for PVM 2.1" on page 440.
3. The PVMSEC CNTRL file is for the Access Security Exits sample code.

Step 15. Release the temporary work minidisk and access the 49E minidisk as file mode A and 39E as file mode B using the following commands:

```
rel a
acc 49E a
acc 39E b
```

Step 16. Regenerate the PVM, DVMUSIE, and EFGVIEW MODULES, and the PVMG LOADLIB using the following commands:

```
pvmbld
mpvmbld
pvmgen
pvmgbld
```

Notes:

1. At normal completion of the PVMGBLD EXEC, you will see RC=8.
2. These commands will copy existing PVM, DVMUSI, and EFGVIEW MODULE, MAP, PRELMAP, and EFGVIEW TEXT files, if they exist on the 49E minidisk, to OMODULE, OMAP, OPRELMAP, and OTEXT files.
3. To regenerate the PVM MODULE with the Access Security Exits included, a control file of PVMSEC must be used with the PVMBLD command. For more information, see "PVMBLD" on page 315.

Step 17. Regenerate the EFGVSTUB MODULE using the following commands:

```
global txtlib dmssvm5
load efgvstub (rldsave origin 20000
genmod efgvstub
```

Step 18. Access the 36E minidisk as file mode B using the following command:

```
acc 36e b
```

Step 19. Rename the existing PVM MODULE, MAP, and PRELMAP files on the 36E minidisk using the following commands:

```
rename pvm module b pvm omodule =
rename pvm map b pvm omap =
rename pvm prelmap b pvm oprelmap =
```

Note: If you have already applied service to PVM, OMODULE, OMAP, and OPRELMAP files may already exist on the 36E minidisk. If these files exist, you can erase them (or save them on a tape).

Step 20. Copy the newly created PVM MODULE, MAP, and PRELMAP files from the 49E minidisk to the 36E minidisk using the following commands:

```
copy pvm module a = = b (olddate
copy pvm map a = = b (olddate
copy pvm prelmap a = = b (olddate
```

Note: You can erase the PVM MODULE, MAP, and PRELMAP files from the 49E minidisk.

Step 21. Access the 38F minidisk as file mode B using the following command:

```
acc 38f b
```

Step 22. Rename the existing EFGVIEW MODULE, MAP, PRELMAP, and TEXT files, and the EFGVSTUB MODULE on the 38F minidisk using the following commands:

```
rename efgview module b efgview omodule =
rename efgview map b efgview omap =
rename efgview prelmap b efgview oprelmap =
rename efgview text b efgview otext =
rename efgvstub module b efgvstub omodule =
```

Note: If you have already applied service to EFGVIEW or EFGVSTUB, OMODULE, OMAP, OTEXT, and OPRELMAP files may already exist on the 38F minidisk. If these files exist, you can erase them (or save them on a tape).

Step 23. Copy the newly created EFGVIEW MODULE, MAP, TEXT, PRELMAP files, and the EFGVSTUB MODULE from the 49E minidisk to the 38F minidisk using the following commands:

```
copy efgview module a = = b
copy efgview map a = = b
copy efgview text a = = b
copy efgview prelmap a = = b
copy efgvstub module a = = b
```

Note: You can erase the EFGVIEW MODULE, MAP, PRELMAP, and TEXT files, and the EFGVSTUB MODULE, from the 49E minidisk.

Step 24. Access the 19E minidisk as file mode B using the following command:

```
acc 19e b
```

Step 25. Rename the existing DVMUSI MODULE, MAP, and PRELMAP files on the 19E minidisk using the following commands:

```
rename dvmusi module b dvmusi omodule =
rename dvmusi map b dvmusi omap =
rename dvmusi prelmap b dvmusi oprelmap =
```

Note: If you have already applied service to DVMUSI, OMODULE, OMAP, and OPRELMAP files may already exist on the 19E minidisk. If these files exist, you can erase them (or save them on a tape).

Step 26. Copy the newly created DVMUSI MODULE, MAP, and PRELMAP files from the 49E minidisk to the 19E minidisk using the following commands:

```
copy dvmusi module a = = b (olddate
copy dvmusi map a = = b (olddate
copy dvmusi prelmap a = = b (olddate
```

Note: You can erase the DVMUSI MODULE, MAP, and PRELMAP files from the 49E minidisk.

Step 27. Access the 38E minidisk as file mode B using the following command:

```
acc 38e b
```

Step 28. Rename the existing PVMG LOADLIB and PVMG LKEDIT files on the 38E minidisk using the following commands:

```
rename pvmg loadlib b pvmg oloadlib =
rename pvmg lkedit b pvmg olkedit =
```

Notes:

1. Initially, there will be no PVMG LKEDIT file on the 38E minidisk.
2. If you have already applied service to PVMG, OLOADLIB and OLKEDIT files may already exist on the 38E minidisk. If these files exist, you can erase them (or save them on a tape).

Step 29. Copy the newly created PVMG LOADLIB and PVMG LKEDIT files from the 49E minidisk to the 38E minidisk using the following commands:

```
copy pvmg loadlib a = = b (olddate
copy pvmg lkedit a = = b (olddate
```

Notes:

1. Initially, there will be no PVMG LKEDIT file on the 38E minidisk.
2. You can erase the PVMG LOADLIB and LKEDIT files from the 49E minidisk.

Step 30. Did you load the optional PVM VM/ESA Help files during PVM installation?

If you **did not** load the optional PVM VM/ESA Help files, you can skip the

next four steps. If you **did** load the optional PVM VM/ESA Help files, continue with the following steps.

Step 31. Reaccess the temporary work minidisk as file mode A using the following command:

```
acc ccuu a
```

Where *ccuu* is the address of the temporary work minidisk.

Step 32. Determine if there are any files on the service tape that affect the optional PVM VM/ESA Help minidisk (19D) using the following command:

```
listfile * help* a
```

If no files are found, you can skip the next two steps. If Help files are found, continue with the following steps.

Step 33. Access the optional PVM VM/ESA Help minidisk as file mode B using the following command:

```
acc 19d b
```

Step 34. Copy the affected files from the temporary work minidisk to the PVM VM/ESA Help minidisk using the following commands:

```
copy * helpabbr a = = b (olddate replace
copy * helppvm a = = b (olddate replace
copy * helpmenu a = = b (olddate replace
copy * helpmsg  a = = b (olddate replace
```

Step 35. Determine if there are any other files on the service tape that you need. There may be replacement files on the service tape to correct problems with files related to installation, service, execs, MPVM macros, or sample files. If, during installation, you loaded these files onto your disks, you will need to load the replacement files from the service tape to those same disks.

The following list shows some of the file types you may see on the service tape. (It is not a complete list. It is only a sample of some of the existing file types.)

$COMP003	$TABLE	ASSEMBLE	CNTRL
CONFSAMP	DIRECT	EXEC	EXECSAMP
FILE	FIRST	GCSSAMP	LKEDCTRL
LKEDIT	LOADLIB	LSEG	MAP
MODULE	MPVM	NAMESAMP	PRELMAP
PSEG	PVMSAMP	SNTENTRY	TEXT
TXTLIB	TXTSEC	VTAMLST	XEDIT

If you find some of these files on the service tape, continue with the following steps. If you do not find any of these files on the service tape, you have completed the manual application of corrective service for text replacement users and can skip the remaining steps.

Step 36. Access all of the minidisks that you used in the previous steps using the following commands:

```
acc 36e b
acc 49e d
acc 38f e
acc 38e f
acc 39e g
acc 29e h
acc 298 i
acc 19e j
acc 193 k
acc 19d l
```

Notes:

1. Do not use file mode A, because it is already being used by the temporary work minidisk.
2. Do not use file mode C, because it is already being used by the 191 minidisk.
3. The commands listed above show a specific order and a specific file mode letter. The order and the file mode letter are not important. You can access these disks using any file mode (except A or C) and in any order. Thus, if any of these file modes are already in use, you can substitute any other file mode letter that is not currently being used.

Step 37. List the files on the temporary work minidisk using the following command:

```
filelist * * a
```

After the command completes, it may be helpful to sort the list by file type using the PF4 key.

Step 38. Scan the files on the temporary work minidisk to find any that you have not already copied. You can ignore the files with a file type of TEXT, TXTSEC, and file types that start with HELP, because you have already handled these files in the previous steps. Also, you can ignore the

- PVM MODULE, MAP, PRELMAP, EXEC, CNTRL, and PVMSEC CNTRL
- DVMUSI MODULE, MAP, PRELMAP and DVMUSIE EXEC
- EFGVIEW MODULE, MAP, PRELMAP, TEXT, EXEC, and MPVM CNTRL
- EFGVSTUB MODULE and TEXT
- PVMG LOADLIB, LKEDIT, LKEDCTRL, and CNTRL
- the rename exec
- the servicing files

If you do find some of these extra files on the temporary work minidisk, continue with the following step. If you do not find any of these extra files on the temporary work minidisk, you have completed the manual application of corrective service for text replacement users and can skip the remaining step.

Step 39. Compare the two files and find the newest one using the following command:

```
filelist fn ft *
```

If the date and time on the file on the temporary work minidisk is the newest version, copy it to the other minidisk using the following command:

```
copy fn ft a = = newfm (olddate replace
```

Where *newfm* is the file mode of the minidisk with the older version of the file.

This completes the manual application of corrective service for text replacement users. For more information about servicing VM, including details of the VMFPLC2 command, see the *VM/ESA Service Guide*, and the *VM/ESA Service Guide for 370* books.

Manually Applying Corrective Service for SOURCE Users

Note: These instructions apply only to the PVM and DVMUSI components. For the EFGVIEW and PVMG components, follow one of the two previous instruction lists.

To complete this job successfully, observe all instructions and commands included in each step. Commands should be entered as shown except where *variable* information is requested.

Step 1. Call your IBM Support Center for the latest service information about the type of service you are installing.

Step 2. Read the letter supplied with the service tape for additional information about this specific tape. You can find a machine readable copy of the letter in Tape File 1 of the service tape.

Step 3. Ensure that all the prerequisites that are already on a PUT tape are applied using VMSERV. The corrective service Memo to Users lists these APARs.

Step 4. Log on to the MAINT virtual machine using the following commands:

```
logon maint
(Enter MAINT's logon password)
ipl cms
```

Step 5. Verify that you have the correct minidisks available:

36E	PVM System minidisk
49E	PVM Text minidisk
39E	PVM Source minidisk
29E	PVM Service minidisk
19E	System Extension Files minidisk
193	IPCS Text minidisk
19D	PVM VM/ESA Help minidisk (optional)

Step 6. Access the 191 minidisk as C using the following command:

```
acc 191 c
```

Step 7. Create a temporary work minidisk and access it as file mode A using the following commands:

```
cp define Tttttt ccuu nnn
format ccuu a
```

Where *tttt* is the temporary work minidisk's device type, *ccuu* is the address of the work minidisk, and *nnn* is the size of the work minidisk in cylinders or blocks. (**Note:** The size of the work minidisk you define will

depend on the amount of service you ordered. If you ordered a lot of service, you should define a large work minidisk.)

Step 8. Mount and attach the PVM corrective service tape to MAINT as 181.

```
cp attach ccuu to maint as 181
cp rewind 181
```

Where *ccuu* is the tape drive real address.

Step 9. Position the tape to the PVM product files using the following command:

```
vmfplc2 scan 5684100A 010101 (eot
```

Step 10. Load the PVM service execs and related files to the temporary work minidisk using the following command:

```
vmfplc2 load * * a
```

This command loads all the files on the tape to your temporary work minidisk.

Step 11. Look for an updated service exec ("5684100A EXC*") on the temporary work minidisk using the following command:

```
listfile 5684100A exc* a
```

If the file is not found, continue with the next step. If the file is found, copy it to file mode C (191 minidisk) and change the file type to EXEC. Then erase it from the temporary work minidisk.

Step 12. Invoke the rename exec supplied on the tape using the following command:

```
5684100A a
```

This exec renames the replacement parts from the tape to the correct file type and erases any service elements shipped on the tape that were not renamed.

Step 13. Access the 29E minidisk as file mode B using the following command:

```
acc 29e b
```

Step 14. List the auxiliary control files and determine which are to be used when building the PVM MODULE.

```
filelist * auxpvm *
filelist * auxsec *
```

If there are multiple copies (the same file exists on two different disks), you should manually check each file (using XEDIT) to determine which one is at the higher level (which file has more APARs listed at the top of the file).

If the higher level file resides on the temporary work minidisk, rename the existing one on the 29E minidisk and copy the file from the temporary work minidisk to the 29E minidisk using the COPYFILE command with the OLDDATE option:

```
rename fn auxpvm b fn newft b
copyfile fn auxpvm a = = b (olddate
rename fn auxsec b fn newft b
copyfile fn auxsec a = = b (olddate
```

Where *fn* is the file name of the file and *newft* is the new file type of the file.

Notes:

1. If this is the first time you are servicing a particular file, you will not find an AUXPVM or AUXSEC file on the 29E minidisk. Therefore, you should copy the file from the work minidisk to the 29E minidisk using the COPYFILE command with the OLDDATE option:

   ```
   copyfile fn auxpvm a = = b (olddate
   copyfile fn auxsec a = = b (olddate
   ```

   Where *fn* is the file name of the file.

2. The AUXSEC files are for updates to the Access Security Exits sample code.

Step 15. Copy (with the OLDDATE and REPLACE options) all update files from the temporary work minidisk to the 29E minidisk using the following command:

```
copy * jnnnnnfm a = = b (olddate replace
```

Where *nnnnn* is the number of an update file on the temporary work minidisk.

Step 16. Access the 49E minidisk as file mode B, the 29E minidisk as file mode D, and the 39E minidisk as file mode E using the following command:

```
acc 49e b
acc 29e d
acc 39e e
```

Step 17. List the text files and determine which are to be used when building the PVM and DVMUSI MODULEs. Note: If you will be reassembling all the updated files, this step is unnecessary and you can skip to the next step. However, the following TEXT files for the PVM component have no source provided, including:

```
DVMMPA TEXT
DVMMPB TEXT
DVMMPC TEXT
DVMMPD TEXT
DVMMPE TEXT
DVMMPF TEXT
DVMMPG TEXT
DVMVTA TEXT
DVMVTB TEXT
DVMVTC TEXT
DVMVTD TEXT
EFGIMA TEXT
EFGIMB TEXT
```

Continue with this step if any of these files exist on the service tape.

```
filelist * text *
filelist * txtsec *
```

If there are multiple copies (the same file exists on two different disks), you should manually check each file (using XEDIT) to determine which one is at the higher level (which file has more APARs listed at the top of the file).

If the higher level file resides on the temporary work minidisk, rename the existing one on the 49E minidisk for TEXT file types, and the 39E minidisk for TXTSEC file types, and copy the files from the temporary

work disk to the appropriate minidisk using the COPYFILE command with the OLDDATE option:

```
rename fn text b fn newft b
rename fn txtsec e fn newft d
copyfile fn text a = = b (olddate
copyfile fn txtsec a = = e (olddate
```

Where *fn* is the file name of the file and *newft* is the new file type of the file.

Notes:

1. The TXTSEC file types are for the Access Security Exits sample code.
2. The following TEXT files, associated with the IPCS facility, should be copied to the 193 minidisk instead of the 49E minidisk:

```
DVMZCO   TEXT
DVMZDS   TEXT
DVMZDV   TEXT
DVMZEX   TEXT
DVMZLI   TEXT
DVMZRO   TEXT
DVMZSY   TEXT
DVMZTK   TEXT
DVMZTR   TEXT
DVMZUS   TEXT
CSIYTXDC TEXT
```

Step 18. Copy any PVM and DVMUSIE EXECs, and PVM CNTRL files from the temporary work minidisk to the 49E minidisk and the PVMSEC CNTRL file to the 39E minidisk using the following commands:

```
copy pvm exec a = = b (olddate replace
copy dvmusie exec a = = b (olddate replace
copy pvm cntrl a = = b (olddate replace
copy pvmsec cntrl a = = e (olddate replace
```

Note

If there are any new or updated COPY or MACRO file types on the corrective service tape, a new macro library must be built for PVM and DVMUSI before any assemblies can be done. To build this new macro library (DVMMAC), use the file called DVMMAC EXEC.

If there are only updates to existing COPY or MACRO file types, the DVMMAC EXEC will not be included on the corrective service tape. If a new exec has not been included, the original exec is valid and there is no need to rename the file. If there are new COPY or MACRO file types added to PVM, the EXEC exists on the temporary work minidisk. Copy the DVMMAC EXEC to the 49E minidisk, and the new COPY or MACRO file types to the 39E minidisk.

Use the following CMS command to build the macro library:

```
vmfmac dvmmac pvm
```

PVM is the name of the control file used for the build. This command will force an update of all the PVM macros and then build the PVM maclib.

Next, use the following commands to put the new PVM maclib on the 49E disk:

```
acc 49e b
copy dvmmac maclib a = = b (olddate replace
```

This whole process is only necessary if there is a source assemble needed.

End of Note

Notes:

1. If your installation has previously installed one of the languages from the national language features tape, you must ensure that the proper languages are specified in the build execs. For more information, see "Step 3. Enabling PVM Components for Additional Languages" on page 233.
2. If your installation is using the Access Security Exits sample code, you must ensure that proper text files are specified in the build execs. For more information, see "Installing and Building the Security Exits for PVM 2.1" on page 440.
3. The PVMSEC CNTRL file is for the Access Security Exits sample code.

Step 19. Access the 49E minidisk as file mode A followed by 29E and 39E. Enter the following commands:

```
acc 49e a
acc 29e b
acc 39e d
```

Step 20. Rename the existing TEXT file types on the 49E minidisk, and TXTSEC file types on the 39E minidisk, for any modules that will be reassembled using the following command:

```
rename * text a = otext =
rename * txtsec d = otxtsec =
```

Note: The TXTSEC file types are for the Access Security Exits sample code.

Step 21. Assemble the modules using the following command:

```
vmfhasm filename pvm
```

Where *filename* is the file name of the module to be assembled.

Notes:

1. Use VMFASM if on any release of VM/SP or VM/ESA 370 feature instead of VMFHASM. Also, make sure the proper maclibs are specified in the PVM CNTRL control file.
2. Assemble files

   ```
   DVMAPA ASSEMBLE
   DVMAPB ASSEMBLE
   DVMAPC ASSEMBLE
   DVMAPD ASSEMBLE
   DVMAPE ASSEMBLE
   DVMAPF ASSEMBLE
   DVMAPG ASSEMBLE
   DVMAPH ASSEMBLE
   ```

   must be assembled on an SP6 (or higher) level of CMS.

3. To assemble the Access Security Exits sample code, use the PVMSEC control file.
4. Any TXTSEC file types should be copied to the 39E minidisk, and erased from the 49E minidisk.

Step 22. Compile the PVM message repositories using the following commands:

```
pvmcmp dvmameng ctl pvm
pvmcmp efgameng ctl pvm
```

Step 23. Regenerate the PVM and DVMUSI MODULEs using the following commands:

```
pvmbld
pvmgen
```

Notes:

1. These commands will copy existing PVM and DVMUSI MODULE, MAP, and PRELMAP files, if they exist on the 49E minidisk, to OMODULE, OMAP, and OPRELMAP files.
2. To regenerate the PVM MODULE with the Access Security Exits included, a control file of PVMSEC must be used with the PVMBLD command. For more information, see "PVMBLD" on page 315.

Step 24. Access the 36E minidisk (PVM 191 minidisk) as file mode B using the following command:

```
acc 36e b
```

Step 25. Rename the existing PVM MODULE, MAP, and PRELMAP files on the 36E minidisk using the following commands:

```
rename pvm module b pvm omodule =
rename pvm map b pvm omap =
rename pvm prelmap b pvm oprelmap =
```

Note: If you have already applied service to PVM, OMODULE, OMAP, and OPRELMAP files may already exist on the 36E minidisk. If these files exist, you can erase them (or save them on a tape).

Step 26. Copy the newly created PVM MODULE, MAP, and PRELMAP files from the 49E minidisk to the 36E minidisk using the following commands:

```
copy pvm module a = = b (olddate
copy pvm map a = = b (olddate
copy pvm prelmap a = = b (olddate
```

Notes:

1. All TXTSEC and $COMP003 file types should also be copied to the 39E disk and erased from the 49E minidisk.
2. You can erase the PVM MODULE, MAP, and PRELMAP files from the 49E minidisk.

Step 27. Access the 19E minidisk as file mode B using the following command:

```
acc 19e b
```

Step 28. Rename the existing DVMUSI MODULE, MAP, and PRELMAP files on the 19E minidisk using the following commands:

```
rename dvmusi module b pvm omodule =
rename dvmusi map b pvm omap =
rename dvmusi prelmap b pvm oprelmap =
```

Note: If you have already applied service to DVMUSI, OMODULE, OMAP, and OPRELMAP files may already exist on the 19E minidisk. If these files exist, you can erase them (or save them on a tape).

Step 29. Copy the newly created DVMUSI MODULE, MAP, and PRELMAP files from the 49E minidisk to the 19E minidisk using the following commands:

```
copy dvmusi module a = = b (olddate
copy dvmusi map a = = b (olddate
copy dvmusi prelmap a = = b (olddate
```

Note: You can erase the DVMUSI MODULE, MAP, and PRELMAP files from the 49E minidisk.

Step 30. Did you load the optional PVM VM/ESA Help files during PVM installation?
If you **did not** load the optional PVM VM/ESA Help files, you can skip the next four steps. If you **did** load the optional PVM VM/ESA Help files, continue with the following steps.

Step 31. Reaccess the temporary work minidisk as file mode A using the following command:

```
acc ccuu a
```

Where *ccuu* is the address of the temporary work minidisk.

Step 32. Determine if there are any files on the service tape that affect the optional PVM VM/ESA Help minidisk (19D) using the following command:

```
listfile * help* a
```

If no files are found, you can skip the next two steps. If Help files are found, continue with the following steps.

Step 33. Access the optional PVM VM/ESA Help minidisk as file mode B using the following command:

```
acc 19d b
```

Step 34. Copy the affected files from the temporary work minidisk to the PVM VM/ESA Help minidisk using the following commands:

```
copy * helpabbr a = = b (olddate replace
copy * helppvm a = = b (olddate replace
copy * helpmenu a = = b (olddate replace
copy * helpmsg  a = = b (olddate replace
```

Step 35. Determine if there are any other files on the service tape that you need.
There may be replacement files on the service tape to correct problems with files related to installation, service, execs, MPVM macros, or sample files. If, during installation, you loaded these files onto your disks, you will need to load the replacement files from the service tape to those same disks.

The following list shows some of the file types you may see on the service tape. (It is not a complete list. It is only a sample of some of the existing file types.)

$COMP003	ASSEMBLE	AUXPVM	AUXSEC
CNTRL	CONFSAMP	COPY	DIRECT
EXEC	EXECSAMP	FILE	FIRST
MACLIB	MACRO	MAP	MODULE
PRELMAP	PVMSAMP	TEXT	TXTSEC

If you find some of these files on the service tape, continue with the following steps. If you do not find any of these files on the service tape, you have completed the manual application of corrective service for source replacement users and can skip the remaining steps.

Step 36. Access all of the minidisks that you used in the previous steps using the following commands:

```
acc 36e b
acc 49e d
acc 39e e
acc 29e f
acc 19e g
acc 193 h
acc 19D i
```

Notes:

1. Do not use file mode A, because it is already being used by the temporary work minidisk.
2. Do not use file mode C, because it is already being used by the 191 minidisk.
3. The commands listed above show a specific order and a specific file mode letter. The order and the file mode letter are not important. You can access these disks using any file mode (except A or C) and in any order. Thus, if any of these file modes are already in use, you can substitute any other file mode letter that is not currently being used.

Step 37. List the files on the temporary work minidisk using the following command:

```
filelist * * a
```

After the command completes, it may be helpful to sort the list by file type using the PF4 key.

Step 38. Scan the files on the temporary work minidisk to find any that you have not already copied. You can ignore the files with a file type of TEXT, TXTSEC, AUXPVM, AUXSEC, and file types that start with HELP, because you have already handled these files in the previous steps. Also, you can ignore the

- PVM MODULE, MAP, PRELMAP, EXEC, CNTRL, and PVMSEC CNTRL
- DVMUSI MODULE, MAP, PRELMAP and DVMUSIE EXEC
- the rename exec
- the servicing files

If you do find some of these extra files on the temporary work minidisk, continue with the following step. If you do not find any of these extra files on the temporary work minidisk, you have completed the manual application of corrective service for source replacement users and can skip the remaining step.

Step 39. Compare the two files and find the newest one using the following command:

```
filelist fn ft *
```

If the date and time on the file on the temporary work minidisk is the newest version, copy it to the other minidisk using the following command:

```
copy fn ft a = = newfm (olddate replace
```

Where *newfm* is the file mode of the minidisk with the older version of the file.

Notes about New Files:

1. If the file is a new ASSEMBLE, COPY, or MACRO file, copy it to the 39E minidisk (if you have one) and specify the OLDDATE option of the COPYFILE command.
2. For any other new files, please refer to the text of the APAR to discover where to copy the file.

This completes the manual application of corrective service for source replacement users. For more information about servicing VM, including details of the VMFPLC2 command, see the *VM/ESA Service Guide*, and the *VM/ESA Service Guide for 370* books.

Automatically Applying Corrective Service

If you are using PVM 2.1.0, you can apply corrective service to your system using the corrective service exec called CORRSERV EXEC made available to you by APAR VM55703.

CORRSERV EXEC Package

Table 18 shows the files that you will need to use the CORRSERV EXEC. These files are found on the MAINT 191 disk, which you must have accessed as **C** and in **R/W** mode.

Table 18. CORRSERV EXEC Package

File	Description
CORRSERV EXEC	Automatically plans for and applies service to your system.
CORRSERV MESSAGES	Contains messages that the CORRSERV EXEC uses for processing.
5684100 BASE	A default file used to build the 5684100 HISTORY file, if you do not already have one. **Note:** The 5684100 BASE file must not be edited.

Table 19 on page 169 shows the files that can be created during processing. These files are created on the MAINT 191 disk.

Table 19. Files Created from CORRSERV Processing	
5684100 HISTORY	Provides the CORRSERV EXEC with: • Location of the PTF, such as: – Tape – Reader – Already loaded and renamed • Whether to build the MODULEs and LOADLIB • Product file identifiers • Parts • Addresses of disks to apply service to • Disks you do not want service applied to
5684100 LOG	Used to log messages while applying a PTF.

5684100 HISTORY File

The CORRSERV EXEC uses the 5684100 HISTORY file to determine what it will process. You are prompted for the necessary information, however, you can manually update the 5684100 HISTORY file. To manually apply updates to the 5684100 HISTORY file, you need to understand its contents.

The 5684100 HISTORY file has three sections:

1. Prompts
2. Parts
3. Disk

Prompts Section

This section begins with the BEGINPROMPTS tag and ends with ENDPROMPTS tag. The variables defined in this section provide the CORRSERV EXEC with the:

- Location of the PTF
- Whether to load MODULEs and LOADLIB from the PTF
- Whether to build the MODULEs and LOADLIB
- The product file prefix
- The product file suffix

Define the prompt tag variables as follows:

corr_srvc_ptf =
Specifies the location of the PTF.

T PTF is on tape
R PTF is in reader
L PTF is already loaded on a disk and has been renamed

pvmg_libuser =
Specifies whether the PVMG LOADLIB will be loaded from the PTF.

'Y' Load the LOADLIB
'N' Do not load the LOADLIB

pvmgbld_libbld =
Specifies whether to build the PVMG LOADLIB.

'Y' Build the LOADLIB
'N' Do not build the LOADLIB

pvm_moduser =
Specifies whether the PVM MODULE will be loaded from the PTF.

'Y' Load the MODULE
'N' Do not load the MODULE

pvmbld_modbld =
Specifies whether to build the PVM MODULE.

'Y' Build the MODULE
'N' Do not build the MODULE

dvmusi_moduser =
Specifies whether the DVMUSI MODULE will be loaded from the PTF.

'Y' Load the MODULE
'N' Do not load the MODULE

pvmgen_modbld =
Specifies whether to build the DVMUSI MODULE.

'Y' Build the MODULE
'N' Do not build the MODULE

efgview_moduser =
Specifies whether the EFGVIEW MODULE will be loaded from the PTF.

'Y' Load the MODULE
'N' Do not load the MODULE

mpvmbld_modbld =
Specifies whether to build the EFGVIEW MODULE.

'Y' Build the MODULE
'N' Do not build the MODULE

efgvstub_moduser =
Specifies whether the EFGSTUB MODULE will be loaded from the PTF.

'Y' Load the MODULE
'N' Do not load the MODULE

efgvstb_modbld =
Specifies whether to build the EFGSTUB MODULE.

'Y' Build the MODULE
'N' Do not build the MODULE

prefix =
Specifies the product file prefix. The prefix for PVM is **J**.

suffix =
Specifies the product file suffix. The suffix for PVM is **FM**.

Note: You can determine the prefix and suffix by looking at the source update file's file type. The first character of the file type is the prefix. The middle characters are the APAR number. The last two characters are the suffix.

Do not remove any of the variables in this section because the CORRSERV EXEC needs them for processing.

Parts Section

This section has a BEGINMODPRTS and ENDMODPRTS tags, and a BEGINPARTS and ENDPARTS tags. The BEGINMODPRTS tag lists files that are tailorable by the user. The BEGINPARTS tag lists the rest of the parts.

Parts are listed starting with either a single colon or with a double colon. The parts starting with a single colon identify the file types of parts for the product. The parts starting with a double colon identify file names and file types of parts targeted to a different disk.

Single colons are the rule, double colons are exceptions to the rule. Therefore, if there are 100 ASSEMBLE parts for the product and 2 of them are assemble files targeted for a different disk the parts section would look like this:

```
::DVM1 ASSEMBLE. = targetdsk
::DVM2 ASSEMBLE. = targetdsk
:ASSEMBLE. = srcdsk
```

Every file for the product must be defined in the parts section.

Note: *targetdsk* can be specified as **'PASS'** for those parts you wish not to service.

Disk Section

The CORRSERV EXEC determines which disks to apply service to using the disk section. This section begins with the BEGINDISK tag and ends with ENDDISK tag.

Define the disk tag variables as follows:

prodsk = 'address'
prodsk = **'PASS'**
Specifies the PVM user's 191 (MAINT's 36E) disk address where service is to be applied . If you specify 'PASS' the disk will not be serviced.

txtdsk = 'address'
txtdsk = **'PASS'**
Specifies the PVM TEXT (MAINT's 49E) disk address where service is to be applied. If you specify 'PASS' the disk will not be serviced.

srcdsk = 'address'
srcdsk = **'PASS'**
Specifies the PVM SOURCE (MAINT's 39E) disk address where service is to be applied. If you specify 'PASS' the disk will not be serviced.

srvdsk = 'address'
srvdsk = **'PASS'**
Specifies the PVM SERVICE (MAINT's 29E) disk address where service is to be applied. If you specify 'PASS' the disk will not be serviced.

systemg = 'address'
systemg = **'PASS'**
Specifies the PVMG user's 191 (MAINT's 38E) disk address where service is to be applied. If you specify 'PASS' the disk will not be serviced.

systemm = 'address'
systemm = **'PASS'**
Specifies the MPVM common (MAINT's 38F) disk address where service is to be applie d. If you specify 'PASS' the disk will not be serviced.

systeme = 'address'
systeme = **'PASS'**
Specifies the MAINT's 19E Y-disk address where DVMUSI service is to be applied. If you specify 'PASS' the disk will not be serviced.

ipcsdsk = 'address'
ipcsdsk = **'PASS'**
Specifies the PVM IPCS (MAINT's 193) disk address where service is to be applied. If you specify 'PASS' the disk will not be serviced.

helpdsk = 'address'
helpdsk = **'PASS'**
Specifies the PVM HELP (MAINT's 19D) disk address where service is to be applied. If you specify 'PASS' the disk will not be serviced.

mnt191 = 'address'
mnt191 = **'PASS'**
Specifies MAINT's 191 disk address where service is to be applied. If you specify 'PASS' the disk will not be serviced.

vtamdsk = 'address'
vtamdsk = **'PASS'**
Specifies VTAM's 191 (MAINT's 298) disk address where service is to be applied. If you specify 'PASS' the disk will not be serviced.

5684100 HISTORY File Requirements

For the CORRSERV EXEC to process, the 5684100 HISTORY file must have:

- BEGIN*section* and END*section* tags at the beginning and end of each section. (*section* being either PROMPTS, PARTS, DISK or MODPRTS).
- Tags within a section begin with :: or : and end with a period.
- When defining a variable, a space must be placed before and after the equal sign. For example,

```
:prodsk. = 'PASS'
```

- Any target disks listed for a part to be copied to in the parts section must exist in the disk section.
- If you specify 'Y' for building a particular part, you cannot have 'PASS' specified for the disks needed to build that part.

Note: The HISTORY file is read to column 30. Anything after column 30 is ignored.

Preparing To Run CORRSERV

__ **Step 1.** Call your IBM Support Center for the latest service information about the type of service you are applying.

__ **Step 2.** Read the letter supplied with the service tape for additional information about this specific tape. You can find a machine readable copy of the letter in Tape File 1 of the service tape.

__ **Step 3.** Log on to the MAINT virtual machine. For example:

```
LOGON MAINT
```

__ **Step 4.** Enter MAINT's logon password.

__ **Step 5.** Enter:

```
IPL CMS
```

__ **Step 6.** Verify that all of the disks specified in the disk section of the 5684100 HISTORY are linked and in R/W mode.

__ **Step 7.** Access the MAINT 191 disk as **C** and **R/W** mode.

__ **Step 8.** Verify that you have the CORRSERV EXEC package.

__ **Step 9.** Verify the location of the PTF.

- If it has been sent electronically, verify that it is in the reader.
- If it is on tape, verify that the tape is mounted and attached as 181 and ensure that it is at the beginning. To do this, enter:

  ```
  CP ATTACH ccuu TO MAINT AS 181
  CP REWIND 181
  ```

- If it is loaded and renamed, verify that it exists on the disk accessed as **A**

Note: You will need to create a temporary work disk if the PTF is not already loaded and renamed. This step is done after the CORRSERV EXEC PLAN stage completes because you will be provided with an estimated size for creating the temporary disk.

Understanding How CORRSERV Processes

This section can help you understand how the CORRSERV EXEC processes both the PLAN and APPLY stages.

PLAN Stage

The PLAN stage:

1. Verifies the CORRSERV EXEC package exists on the MAINT 191 disk.
2. Verifies that you have MAINT 191 disk accessed as **C** and in **R/W** mode.
3. Creates a 5684100 HISTORY file, if you do not already have one.
4. Gives you a chance to XEDIT the 5684100 HISTORY file.
5. Checks the 5684100 HISTORY file for errors. If there are errors, the error messages are both displayed and logged into a 5684100 LOG file. Here, if there are errors, you are asked if you want to XEDIT the 5684100 HISTORY file to correct these errors.
6. Checks for a new 5684100 BASE file comparing it with the 5684100 HISTORY file. If anything is different, you are notified and prompted for verification of a new tag and target.
7. Checks the size of the source update files and the rename exec's (5684100 EXEC) renamed parts shipped with the PTF and determines an estimated size to be used when creating your temporary disk.
8. Notifies you of the temporary disk estimate in CMS block size and logs it into the 5684100 LOG file.

Creating Your Temporary Disk

After the PLAN stage completes processing, create a temporary disk using the estimated size the PLAN stage provided for you and access it as file mode **A**. For example:

```
CP DEFINE Tttttt ccuu nnn
FORMAT ccuu A
```

The *tttt* is the temporary disk device type. The *ccuu* is the address of the temporary disk. The *nnn* is the size, which was given to you by the PLAN stage.

Note: If you have CKD type DASD devices, you will need to convert the CMS block size to cylinders.

APPLY Stage

The APPLY stage:

1. Verifies that the CORRSERV EXEC files exist on the MAINT 191 disk.
2. Verifies that the MAINT 191 disk is accessed as **C** and in **R/W** mode.
3. If the PTF is in the reader or on tape, it:
   a. Verifies that you have the temporary disk accessed as **A**.
   b. Reads the 5684100 HISTORY file to determine processing.
   c. Loads the appropriate parts and source onto the temporary disk.
   d. Renames as appropriate.
4. Compares dates to determine the newer part.

   For AUX files, newer parts are determined by the one having the most entries. For TEXT files, newer parts are determined by the one having the most updates applied to it.
5. Copies newer files from the temporary disk to the disk defined in the disk section of the 5684100 HISTORY file. A message is sent to the 5684100 LOG file stating which, if any, files did not get copied.
6. If specified in the 5684100 HISTORY file, builds the MODULEs and LOADLIB.

Note: If you have user modifications on your system you may want to verify which parts were applied by doing a manual compare.

This completes the automatic application of corrective service.

Use this format to begin the CORRSERV EXEC processing.

CORRSERV

Purpose

Gets a PTF from your reader, a tape, or a disk and applies it to your system.

Operands

PLAN
Specifies you want to begin the planning stage.

APPLY
Applies the PTF to your system.

Usage Notes

1. It is required that you use the PLAN operand before the APPLY operand to provide you with a certain amount of planning verification.

Location of PVM's Files on MAINT's Minidisks

The following table outlines the locations of PVM's files on MAINT's minidisks. This list may be superseded as files are added to PVM. If you discover such a file, look for similar files to determine the location for the new file.

Table 20 (Page 1 of 3). Location of PVM files on Maint's Minidisks

Minidisk	File
191	I5684NLS EXEC I5684100 EXEC MPVM SNTENTRY MPVMP SNTENTRY PVMGSAMP DIRECT PVMSAMP DIRECT READTHIS FIRST
36E	PROFILE PVMSAMP PROFPVM EXECSAMP PVM CONFSAMP PVM MAP PVM MODULE PVM PRELMAP RUNPVM EXEC UCOMDIR NAMESAMP
19E	DVMUSI MAP DVMUSI MODULE DVMUSI PRELMAP MPVM EXEC PASSTHRU EXEC PVMSNAP EXEC

Table 20 (Page 2 of 3). Location of PVM files on Maint's Minidisks

Minidisk	File
39E	MPVMBLD EXEC PVMBLD EXEC PVMDEST FILE PVMG LKEDCTRL PVMGBLD EXEC PVMSEC CNTRL SNAMODET ASSEMBLE All **$COMP003** file types All **$EXEC** file types All **ASSEMBLE** file types, except All **COPY** file types, except DVMMAC COPY All **MACRO** file types All **REPOS** file types All **TXTSEC** file types All **UPDTSEC** file types
29E	All **AUXPVM** file types All **AUXSEC** file types All **J*nnnnn*FM** file types
49E	AMENG SCREENS CMSINST MPVMLIST DISPLAY EXEC DVMMAC MACLIB DVMMAC COPY DVMMAC EXEC DVMUSIE EXEC EFGDCSS EXEC EFGVIEW EXEC EFGZAP EXEC PVM EXEC PVMCMP EXEC PVMGEN EXEC All **CNTRL** file types, except PVMSEC CNTRL All **LSEG** file types All **PSEG** file types All **TEXT** file types, except: CSIYTXDC TEXT DVMZCO TEXT DVMZDS TEXT DVMZDV TEXT DVMZEX TEXT DVMZLI TEXT DVMZRO TEXT DVMZSY TEXT DVMZTK TEXT DVMZTR TEXT DVMZUS TEXT EFGVIEW TEXT
19D	All **HELPABBR** file types All **HELPMENU** file types All **HELPMSG** file types All **HELPPVM** file types **Note:** Minidisk 19D is for the American English help files, minidisk 401 is for the Kanji help files, minidisk 402 is for the upper case English help files, and minidisk 405 is for the German help files.

Table 20 (Page 3 of 3). Location of PVM files on Maint's Minidisks

Minidisk	File
193	CSIYTXDC TEXT DVMZCO TEXT DVMZDS TEXT DVMZDV TEXT DVMZEX TEXT DVMZLI TEXT DVMZRO TEXT DVMZSY TEXT DVMZTK TEXT DVMZTR TEXT DVMZUS TEXT HELP PVM
38E	PROFILE GCSSAMP PVMG LOADLIB PVMG LKEDIT PVMG CONFSAMP RUNPVMG EXEC
38F	$CMSINST EXEC EFGVCMDT EXEC EFGVCMDT $TABLE EFGVCMDT AMENG EFGVIEW MODULE EFGVIEW MAP EFGVIEW PRELMAP EFGVIEW TEXT EFGVNLS MASTER EFGVSTUB MODULE MPVM COMDDISK MPVM NAMESAMP MPVMSYS NAMESAMP All **MPVM** file types All **XEDIT** file types
298	PVMG VTAMLST SNAMODET ASSEMBLE

Chapter 12. Collecting Problem Information

DMTI

Diagnosis, Modification, and Tuning Information is provided to help you perform diagnosis tasks of PVM and its components.

WARNING: Do not use this Diagnosis, Modification, and Tuning Information as a programming interface.

This chapter points out what problem determination aids are available for use with PVM, PVMG, and MPVM, and how these aids are invoked. The problem determination aids are intended for use by:

- IBM Support Personnel
- System programmers with a knowledge of PVM structure
- System administrators and operators with the assistance of IBM Support Personnel.

Dumps and traces are PVM problem determination aids. Dumps provide information about PVM when an error causes PVM or a component of PVM to abnormally terminate. Dumps are automatically produced when PVM or the component fails.

Traces are used when there are symptoms of a PVM problem but PVM continually recovers. In these cases, dumps are not produced, and traces provide information used to solve a problem. A system administrator or operator has to issue commands to PVM to produce traces.

Some types of problems you may encounter with PVM include:

- Users can't access a target system.

  Check your PVM configuration file to make sure that the target system is set up and defined correctly. You may also have hardware problems preventing the connection to the target system.

- PVM has very slow response time.

  Check to make sure that all of the PVM performance options are being used. "Enhancing PVM Performance" on page 199 describes these performance options.

- PVM has stopped running or abnormally terminated.

  Use the PVM trace and dump facilities to try to find the problem.

If you cannot find a solution to a problem, you need to contact IBM Support Personnel. Be prepared to answer questions about:

- The release and service levels of the VM system that PVM is running in
- Other products that are running in your VM system
- The release and service level of your PVM
- Your PVM configuration.

You may want to create a table of your network, similar to Table 7 on page 65, for your PVM configuration. Having this information available for the IBM Support Personnel can sometimes help get your PVM problems fixed more quickly.

© Copyright IBM Corp. 1991,1993

Note: Help for all PVM messages with identifiers is available through the VM/ESA HELP Facility. Only messages without identifiers, PVM/IPCS support messages, and Compiler messages are documented in this book.

Trace Facility for the PVM Component

PVM provides tracing for:

- Line driver I/O
- Line driver data transfer
- Session or MPVM user data transfers
- System functions.

Some forms of the TRACE command can be issued before a line starts. The type of tracing specified by the TRACE options is invoked when the line is started at a later time. These TRACE commands can be placed in the PVM profile or can be entered from the PVM user ID:

- TRACE LINE IO
- TRACE LINE DATA
- TRACE LINE WRAP
- TRACE NODE IO
- TRACE NODE DATA
- TRACE NODE WRAP.

(See page 334 for the exact syntax for the TRACE command.)

Later, if a line develops a problem, having TRACE invoked in this way can make it easier to determine the source of that problem.

To use the TRACE command with the SYSTEM ETRACE options, PVM Version 2 must be running on VM/SP Release 6 or VM/ESA Release 1.0 (370 Feature).

See Appendix C, "PVM Commands" for a description of the TRACE command and its options.

System Services Tracing

System services tracing on an IUCV diagnose function is performed once control is returned to the support task upon an IUCV diagnose function, and a condition code has been saved. The event will be recorded in the PVM system trace. This information will include the IUCV function type, user ID and path ID of the MPVM user, and the length of data sent or received from the MPVM user. If the IUCV diagnose function ends with an error, a PVM message is produced to record the error.

TRACE USER Command

Data for one or all user sessions can be traced using PVM's TRACE USER command. See "TRACE" on page 334 for information about the PVM TRACE command.

If the *vmid* is a VM user ID of a MPVM user, the data for all of that user's sessions will be traced if the *nodeid* is the origin node ID of the MPVM user. Data for one session can also be traced by specifying the destination user ID and node ID in the nodeid and vmid parameters of the TRACE parameters.

Tracing Line Driver I/O Activity

When the IO and ON options of the TRACE command are used, line driver I/O activity tracing begins automatically when the associated line driver is started, and continues as long as the line driver is active. When the WRAP option is used, this information is recorded in an internal wraparound trace table. If the line driver terminates abnormally because of an unexpected line problem, the line driver uses the CP DUMP command to automatically print out the trace table contents. This includes a record of the events occurring on that line just prior to the failure.

The format of each entry of the trace table when the WRAP option is used is as follows:

Table 21. Format of Trace table entries

Offset	Length	Contents
0	8	CSW
8	1	Sense byte (if CSW indicates UC)
9	3	CAW
12	4	Identifier (NBSC, NCTC, RSF, SIM, C381, or C382)
16	8	First CCW
24	16	Data associated with first CCW
40	8	Second CCW
48	16	Data associated with second CCW
64	8	Third CCW
72	16	Data associated with third CCW
88	8	Fourth CCW
96	16	Data associated with fourth CCW

Note: Each entry contains at least 1 and up to 4 CCWs.

A different trace table format is used by the 3088 line driver when attached only to the ESCON channel. Two sense bytes are returned on a unit check, and this trace table accommodates those two sense bytes.

The format of each entry of the ESCON channel trace table when the WRAP option is used is as follows:

Table 22 (Page 1 of 2). Format of ESCON CTC Trace table entries

Offset	Length	Contents
0	8	CSW
8	4	CAW
12	4	Identifier (S38R, S38W)
16	2	Sense bytes
18	8	First CCW
26	18	Data associated with first CCW
42	10	Second CCW
50	18	Data associated with second CCW
66	10	Third CCW
74	18	Data associated with third CCW
90	10	Fourth CCW

Table 22 (Page 2 of 2). Format of ESCON CTC Trace table entries		
Offset	**Length**	**Contents**
98	18	Data associated with fourth CCW
Note: Each entry contains at least 1 and up to 4 CCWs.		

Use the LINE or NODE options of the TRACE command whenever there appears to be a communication problem on a PVM link. All I/O activity on the line is recorded and indicates problems in teleprocessing hardware and protocol. Interpreting TRACE results requires a knowledge of IBM System/370 channel commands and basic teleprocessing protocol.

When you use the IO or DATA options of the TRACE command, output is written to a virtual printer that you specify. When you use the WRAP option, output is placed in a wraparound trace table. For more information on using the TRACE command, see page 334.

Line Driver Data Transfer Tracing

Chronology and contents of data buffers passed to and from a line driver can be traced by using the TRACE command with the LINE and DATA, or NODE and DATA options. This is useful when a problem is suspected with a data transfer between PVM components or between systems in a network. The line driver dumps the contents of each data buffer it sends or receives across the telecommunication line. The duration of the trace is controlled by the ON and OFF parameters of the TRACE command. The TRACE command is described in Appendix C, "PVM Commands."

PCCF Tracing

PCCF activity can be recorded by using the TRACE PCCF command or by using the START PCCF command with the TRACE option. You can start PCCF tracing only for active PCCF tasks. To stop a PCCF trace, use the OFF parameter of the TRACE command. The TRACE command is described in Appendix C, "PVM Commands."

Tracing System Activity

Use the TRACE command with the SYSTEM option to assist in problem source determination when internal PVM trouble is suspected. This combination issues a chronological trace of the activities of key PVM system components. The trace facility records requests for service, passed data (if any), execution of services, and status information. The sequence of the trace records is also a map that shows the transfer of control between various components.

The output resulting from the TRACE command can be directed to the PVM user ID, to a virtual printer, or to a CPTRAP file. The output, however, can be lengthy so it often cannot be displayed on the PVM user ID's display station. The TRACE command is described in Appendix C, "PVM Commands."

Entering Trace Records into the CPTRAP File

You can have system trace data written to a CPTRAP file by using the TRACE command with the SYSTEM and ETRACE options. When ETRACE is set ON, PVM produces system trace records that are identical with the information written to the console or virtual printer by the TRACE command with the SYSTEM and CONSOLE options or the TRACE command with the SYSTEM and PRT options. These records are written to CP and saved in a CPTRAP managed reader file. CPTRAP support includes the writing of time stamp records whenever the file is started, each time a data block is written to the CPTRAP file, and when the file is stopped. For more information on the CPTRAP facility, see the *VM/ESA: Diagnosis Guide, LY24-5250*. For the format of the CPTRAP command, see the *VM/ESA: CP System Command Reference for 370, SC24-5434*.

If you are running on a VM/ESA (370 Feature) or VM/SP system, you can use the IPCS IPCSSCAN command to format the trace records for display or printing. For more information about the IPCSSCAN command and its subcommands, see the *VM/ESA: Interactive Problem Control System Guide and Reference for 370, SC24-5438*.

To use the ETRACE system trace, follow these steps:

1. From a class E user ID, like MAINT or the system operator, enable PVM for writing records to the CPTRAP file and start the CPTRAP function. Enter these CPTRAP commands:

   ```
   cptrap id trapid set trapset type gt allowid userid 3e
   cptrap to userid
   cptrap enable id trapid
   ```

2. Next, from the PVM user ID, enter:

   ```
   trace system etrace on
   ```

   to tell PVM to write system trace records to CPTRAP.

3. To terminate PVM system tracing, enter from the PVM user ID:

   ```
   trace system etrace off
   ```

   Then, from the class E user ID, enter:

   ```
   cptrap drop set trapset
   ```

   to terminate CPTRAP and create a reader file.

4. From the class E user ID, enter:

   ```
   ipcsscan spool spoolid cptrap
   select 3e machtype 03
   trace for count format
   ```

   This accesses the CMS disk file containing the PVM system records. IPCSSCAN calls PVM to process the PVM records.

Types of PVM Trace Records

PVM tasks produce trace records when they request PVM services. These trace records are described as follows in alphabetic order.

---

macrocall **FOR TASK APG** *tcbid* **PATH** *hhhh* **CC**=*cc* **RC**=*rc*

Produced By: The APPC/VM line driver task

When: During processing of an IUCV or APPC/VM macro call.

---

task1 **DETACHED BY TASK** *task2 tcbid*

Produced By: The DETACH SVC routine.

When: It detaches *task1* as requested by *task2*.

---

type **SVC TASK** *task tcbid*

Produced By: The SVC interrupt handler

When: It receives the initial request for *type* service from *task*. When the requested service has been carried out, the corresponding SVC routine produces a trace entry record.

APPC CONNECT FOR TASK *task tcbid* **PATH** *nnnn* **CC**=*cc* **RC**=*rc*

Produced By: The IUCV/APPC SVC handler

When: An APPC/VM CONNECT request has been made for *task*. The path, condition code, and return code from that request are displayed.

EXCP TASK *task tcbid* **DEV**=*device* **CC**=*cc* **CAW**=*addr*

Produced By: The EXCP SVC routine.

When: It initiates the I/O operation requested by *task* to *device* and the I/O operation completed with a zero condition code.

EXCP TASK *task tcbid* **DEV**=*device* **CC**=*cc* **RC**=*rc* **CSW**=*hhhhhhhh hhhhhhhh*

Produced By: The EXCP SVC routine.

When: It initiates the I/O operation requested by *task* to *device* and the I/O operation completed with a nonzero condition code. Condition code *cc* indicates the success or failure on initiating the I/O; return code *rc* provides additional information if unsuccessful, as does the information in the CSW.

EXCP TASK *task tcbid* **DEV**=*device* **CHANNEL BUSY**

Produced By: The EXCP SVC routine.

When: It initiates the I/O operation requested by *task* to *device*, but the channel to the device busy with other activity.

IUCV CONNECT FOR TASK *task tcbid* **PATH** *nnnn* **CC**=*cc* **RC**=*rc*

Produced By: The IUCV/APPC SVC handler

When: An IUCV CONNECT request has been made for *task*. The path, condition code, and return code from that request are displayed.

IUCV SEVER FOR TASK *task tcbid* **PATH** *nnnn* **CC**=*cc* **RC**=*rc*

Produced By: The IUCV/APPC SVC handler

When: An IUCV SEVER request has been made for *task*. The path, condition code, and return code from that request are displayed.

IUCV *type* **USER** *task tcbid* **PATH** *nnnn* **CC**=*cc* **RC**=*rc* **LOC**=*addr* **LEN**=*len*

Produced By: The user task handler for *tcbid*

When: An IUCV request of *type* has been made by a task. The path, condition code, and return code from that request are displayed along with the address and lenth of any data sent with the request.

IUCV *type* **TASK** *task tcbid* **PATH** *nnnn* **CC**=*cc* **RC**=*rc* **LOC**=*addr* **LEN**=*len*

Produced By: The IUCV task to the PVMG virtual machine

When: An IUCV request of *type* has been made by a task. The path, condition code, and return code from that request are displayed along with the address and lenth of any data sent with the request.

LDEV *addr* **TERMINATED BY CP**

Produced By: The external interrupt handler

When: It receives an external interrupt indicating that CP has terminated a logical device.

LDSF CREATE FOR TASK *task tcbid* **LDEV** *addr* **CC**=*cc* **RC**=*rc*

Produced By: The logical device create routine

When: A DIAG 7C CREATE subfunction has been invoked for *task*. The logical device *addr*, condition code, and return code from that request are displayed.

SMSG BUFFER ALLOCATED AT LOCATION *addr* **LEN** *len*

Produced By: The external interrupt handler

When: An SMSG buffer was allocated using the DMSFREE macro.

SMSG BUFFER RELEASED AT LOCATION *hhhhhh* **LEN** *len*

Produced By: The command processor task

When: Completion of processing of SMSG data.

TASK *task tcbid* **ALLOCATED** *nn* **SEGMENT BUFFER AT** *hhhhhh*

Produced By: The buffer manager task

When: Either the DVMGETL or DVMGETS macro is issued by *task* to allocate a segment from the buffer pool.

TASK *task tcbid* **BUFFER MANAGER ERROR** ******

Produced By: The buffer manager task

When: The buffer manager has aborted. An associated storage dump was taken.

TASK *task tcbid* **CONVERTED TO** *nn* **SEGMENT BUFFER AT** *hhhhhh*

Produced By: The buffer manager task

When: The DVMCBUF macro is issued by *task* to change a 4096 byte buffer to a smaller buffer.

TASK *task tcbid* **DMSFREE LOCATION** *hhhhhh* **LEN**=*len*

Produced By: A PVM common function subroutine called to record the storage area allocated by DMSFREE.

When: The DMSFREE macro was issued by a PVM function.

TASK *task tcbid* **DMSFRET LOCATION** *hhhhhh* **LEN**=*len*

Produced By: A PVM common function subroutine called to record the storage area released by DMSFRET.

When: The DMSFRET macro was issued by a PVM function.

TASK *task tcbid* **FLAG**=*hhhhhhhh* **WAIT DROPPED THROUGH**

Produced By: The WAIT SVC routine

When: *task* is not placed on the wait queue, as requested, because at least one of its required services or events indicated in *hhhhhhh* has already completed.

TASK *task tcbid* **POSTED** *interrupttype interruptelement*

Produced By: The external interrupt handler

When: An external interrupt was received for *task*. The type of interrupt, that is, VMCF or LDSF, and the contents of the task interrupt element are displayed.

TASK *task tcbid* **POSTED IO** *device* **CSW** *hhhhhhhh hhhhhhhh* **CAW** *addr*

Produced By: The I/O interrupt handler

When: It receives an I/O interrupt to *task* from *device*. The entry also includes the channel status word (CSW) and channel address word (CAW) information in hexadecimal notation.

TASK *task tcbid* **POSTED IUCV** *interrupt element*

Produced By: The external interrupt handler

When: An external IUCV interrupt was received for *task*. The contents of the task interrupt element are displayed.

TASK *task tcbid* **PROGRAM CHECK PSW** *hhhhhhhhhhhhhhhh*

Produced By: The program interrupt handler

When: A program interrupt has occurred. *task* was executing at the time of the interrupt. The Program Status Word (PSW) is presented.

TASK *task tcbid* **RELEASED** *nn* **SEGMENT BUFFER AT** *hhhhhh*

Produced By: The buffer manager task

When: The DVMFREB macro is issued by *task* to release a buffer.

TASK *task tcbid* **RESUMED**

Produced By: The dispatcher

When: The previously running *task* is dispatched.

TASK *task tcbid* **WAITING FLAGS**=*hhhhhhhh*

Produced By: The WAIT SVC routine

When: *task* is placed on the wait queue, as requested, until required services or events indicated in *hhhhhhhh* are completed.

TASK *task1 tcbid* **POSTED BY** *task2 tcbid* **FLAG** *hhhhhhhh*

Produced By: The POST SVC routine

When: It posts *task1* as requested by *task2* with information *hhhhhhhh*.

TASK *task1 tcbid* **RELEASED ALL BUFFERS FOR TASK** *task2 tcbid*

Produced By: The buffer manager task

When: The DVMFREB macro is issued by *task1* with the *release all* option to release all buffers allocated to *task2*.

TASK *task1 tcbid* **RNG-- POSTED BY** *task2* **FLAG**=*hhhhhhhh* **NP**=*hhhhhh...*

Produced By: The dispatcher

When: It dispatches *task1* that was waiting for the service of the event completed by *task2* as indicated by the information *hhhhhhhh*. NP= reflects the first 48 bytes of the Post Queue Element (PQE) argument list passed in posting.

TASK *task1* **ATTACHED BY TASK** *task2 tcbid*

Produced By: The ATTACH SVC routine

When: It attaches *task1* as requested by *task2*

TASK *tcbid* **LDEV** *hhhh* **ACCEPT CC**=*cc* **RC**=*rc* **LOC**=*addr* **LEN**=*len*

Produced By: The logical device support task

When: Following invocation of DIAG 7C ACCEPT subfunction.

TASK *tcbid* **LDEV** *hhhh* **PRESENT CC**=*cc* **RC**=*rc* **LOC**=*addr* **LEN**=*len*

Produced By: The logical device support task

When: Following invocation of DIAG 7C PRESENT subfunction.

TASK *tcbid* **LDEV** *hhhh* **STATUS CC**=*cc hhhh* **CC**=*cc* **RC**=*rc*

Produced By: The logical device support task

When: Following invocation of DIAG 7C STATUS subfunction.

TASK *tcbid* **LDEV** *hhhh* **TERMINATED**

Produced By: The logical device support task

When: Following invocation of DIAG 7C TERMINATE subfunction.

TASK *tcbid* **DIAG98 LOCK UNLOCK V=** *vvvvvv*

Produced By: The 3088 line driver task

When: During unlocking of a page using DIAG 98.

TASK *tcbid* **DIAG98 LOCK V=** *vvvvvv* **R=** *rrrrrr*

Produced By: The 3088 line driver task

When: Completion of the locking of a page using DIAG 98.

TASK CON IUCV RECEIVE *MSG PATH *nnnn* **CC=** *cc* **RC=** *rc*

Produced By: The command processor task

When: A message system service interrupt has occurred.

TCB ALLOCATED FOR TASK *task1* **AT LOCATION** *hhhhhh* **LEN** *nnnn*

Produced By: The ATTACH SVC routine

When: It allocates an area for a TCB control block.

TCB FRETTED FOR TASK *task* **AT LOCATION** *hhhhhh* **LEN** *len*

Produced By: The DETACH SVC routine.

When: The TCB storage is returned using the DMSFRET macro.

TIMER POST TASK= *task tcbid* **TIME=** *time*

Produced By: The external interrupt handler

When: It receives a timer interrupt to *task*. The entry also reflects the time.

VMCF FUNCTION *nn* **USER** *vmid* **MID=** *id* **RC=** *rc* **LOC=** *addr* **LEN=** *len*

Produced By: The PVM-CMS support task

When: The DIAG 68 (VMCF) is issued to communicate with user *vmid*.

The Dump Facility

The PVM component does basic error checking during its processing. If a problem, such as a program check, causes PVM to abnormally terminate, the system produces a full dump of the type specified in the DUMP record of the configuration file. Abnormal termination of some PVM component parts cause partial dumps. These dumps are automatically produced for you by PVM. Comment fields of the dump output include date and time stamps in the following format:

```
mm/dd/yy hh:mm:ss
```

In addition to producing a dump, PVM writes informational messages to the PVM user ID and to the optional audit file (if the AUDIT record is included in the configuration file).

Full Dumps

Multitasking Supervisor

The PVM multitasking supervisor does basic checking during its processing. If it detects a situation that makes it impossible to continue, it issues a multitasker abort (DVMABRT macro with a special error code parameter). Error message ABR000E (with corresponding error code) is issued, a dump (of the type specified in PVM CONFIG file) is produced for nonzero return codes, and control returns to CMS.

Most invalid SVC parameter lists cause an error return code to the calling task.

Any program check within PVM results in a full virtual storage dump. PVM tries to continue processing without the aborted task. If that is not possible, PVM aborts.

Session Manager

The session manager issues error messages for invalid commands, and terminates the processing of an event if it cannot perform that task. It then waits for the next event.

Buffer Manager

The buffer manager aborts if it cannot get initial free storage to build the buffer pool, if it finds an internal error within the buffer pool, if a task releases a buffer belonging to another task, or if the (passed) buffer address is invalid. The buffer manager abort procedure is as follows:

- Issue the error message describing the problem
- Store the current registers in the TREGs field of the current task control blocks (TCB)
- Request the appropriate type dump
- The aborted task is suspended
- PVM tries to continue processing without the suspended task.

Partial Dumps

Line Drivers

PVM line drivers check telecommunication line protocol and attempt to recover from errors. If error recovery fails, the line driver dumps its TCB along with its internal (wraparound) trace table, and then terminates. The session manager restarts the line driver if a connection was previously established.

The networking line drivers maintain a block sequence counter and restart the connection if a block is found to be missing.

The line drivers keep track of all errors and writes the error statistics to the PVM user ID and to the optional audit file at termination time.

User, Printer, and PCCF Support Tasks

These PVM tasks check for error conditions and terminate the session, or sessions in the case of an MPVM user, if any error cannot be handled. Such an error causes a dump of related control blocks and associated buffers as well as termination of the task.

Analyzing Dump Information

You can use the IPCS component of VM/ESA (370 Feature) and VM/SP, or the Dump Viewing Facility of VM/ESA and VM/XA to analyze system failures and then manage problem information and status. Using these facilities, you can interactively retrieve and examine information from PVM storage dumps to help track and report PVM problems. IPCS can also be used to examine the trace records in a CPTRAP reader file. For more information on creating a CPTRAP file in PVM, see page 182.

IPCS and the Dump Viewing Facility also process dumps created by CP and CMS; however, this book gives only an overview of how these facilities operate with PVM and tells how you can obtain information using IPCS commands. The Dump Viewing Facility supports obtaining information from PVM dumps by supporting some IPCS commands. IPCS commands that apply only to PVM are described in the section "PVM/IPCS Support Commands" on page 339. For details about general IPCS operation and use, see the *VM/ESA: Interactive Problem Control System Guide and Reference for 370, SC24-5438*. For details about general Dump Viewing Facility operation and use, see the *VM/ESA Dump Viewing Facility Operation Guide and Reference SC24-5530* and the *VM/XA: System Product Dump Viewing Facility Operation Guide and Reference*, SC23-0359.

Creating A Problem Report

To create a problem report using IPCS:

1. You need a VMDUMP type dump. To get a VMDUMP for a problem you can either:
   - Put a DUMP record with the VMDUMP option in the PVM configuration file. The VMDUMP type dump is created when a problem causes PVM to abnormally terminate. Or,
   - Issue the CP VMDUMP command when a problem is suspected (the dump's file type must be 'PVM'). The CP VMDUMP command is discussed fully in *VM/ESA: CP General User Command Reference for 370, SC24-5433*.
2. You need a compressed load map of the CMS nucleus and PVM load maps. Use the MAP command to create a compressed load map of the CMS nucleus and PVM load maps. (For a description of the MAP command, see "MAP" on page 344.)
3. Enter:

```
ipcsdump
```

The IPCSDUMP command creates a problem report, attaches the compressed load map to the dump, and places the dump on disk. The report is given the name PRB*nnnnn* REPORT and the dump placed on the disk is named PRB*nnnnn* DUMP. (*nnnnn* is a problem number assigned to the problem by IPCS.) IPCSDUMP also adds the new data to a historical problem file and compares it against existing data in the file for any duplicate problems. The problem report contains keyword data and text data that characterize the problem.

For more information on the IPCSDUMP command and on the contents of the problem report it creates, see *VM/ESA: Interactive Problem Control System Guide and Reference for 370, SC24-5438*.

Examining the Dump

After you have created a problem report with IPCSDUMP, you can interactively locate, display, print, or otherwise examine data in the dumps by using the IPCSSCAN command. IPCSSCAN uses the PRB*nnnnn* DUMP file created by the IPCSDUMP command. IPCSSCAN is also used to view records in the CPTRAP file created by PVM while the TRACE SYSTEM ETRACE command was ON.

To initiate IPCSSCAN, enter:

```
ipcsscan
```

You will be prompted for the number of the PRB*nnnnn* DUMP. See page 342 for a description of the IPCSSCAN command and its format.

After initiating IPCSSCAN, you can make use of subcommands provided by the PVM/IPCS support and by IPCS itself. After you issue IPCSSCAN, you can issue the following PVM-supported subcommands:

COMBLOCK Displays the DVMCOMN control block.

DEVQUE Displays the active logical device support task descriptor blocks (DVMAATs) from the list associated with a particular node (LINK) or for all nodes defined to the PVM system.

HELP Displays a list of the available common and PVM/IPCS IPCSSCAN subcommands and their explanations.

LINKS Displays the link descriptor block (DVMALT) for a particular node (LINK) or for all nodes defined to the PVM system.

ROUTES Displays the route table that was in use by the PVM system.

SYSBLOCK Displays the DVMSYSB control block.

TASKQUE Displays the task control blocks (DVMTCBs) of those tasks on the PVM system's run queue.

USERQUE Displays the active user support task descriptor blocks (DVMAUTs) from the list associated with a particular node (LINK) or for all nodes defined to the PVM system.

See "PVM/IPCS Support Commands" on page 339 for descriptions and output samples of the PVM IPCSSCAN subcommands.

Other PVM Problem Determination Actions

PVM's MPVM user support task will do basic error checking during its normal processing. If a problem occurs, the MPVM support task produces a dump to the trace printer. Sessions and the user task(s) will be terminated when unrecoverable errors occur at the session or task level. Information useful for IBM service personnel is included in the dump that results.

Tracing MPVM

If you encounter a problem with MPVM, you can trace MPVM's EFGVIEW module. You can trace the module at a certain point in its processing, or from the time the module is first invoked.

Tracing Pieces of EFGVIEW Execution

If users are encountering problems with the MPVM user module (EFGVIEW), you can set on MPVM trace variables to help understand what processing is taking place. These variables are of the form TRACE*n*, where *n* is a number from 1 to 7 (8 — 10 are reserved). Each TRACE*n* variable corresponds to a particular type of tracing operation.

In the trace output file for EFGVIEW, MPVM uses keywords to categorize the tracing information being reported. Here are the types of tracing you can do:

Number	Keyword	Type of Debug Performed
1	FLOW	Shows entry into and return from every routine
2	STATUS	Gives status information at various critical decision points
3	MESSAGE	Logs all messages
4	BUFFER	Reports storage buffer management statistics
5	IUCV	Shows IUCV parameter lists and data flow
6	CONSOLE	Shows all console interrupts and commands
7	IMAGE	Step through processing of the MPVM routine that handle screen images
8	Reserved	
9	Reserved	
10	Reserved	

Note: The TRACE output may contain sensitive data, including passwords.

MPVM provides three shortcuts for turning all currently-defined TRACE*n* variables for EFGVIEW on or off:

1. The MPVM macro TRACES
2. A ?TRACES parameter on the MPVM command
3. A CMS global variable. You can issue the command:

   ```
   GLOBALV SELECT MPVM SETP TRACE ON
   ```

For more information on tracing the EFGVIEW user interface module, see the *VM/Pass-Through Facility Programming Reference*.

The EFGVIEW module checks, at various points in its processing, which (if any) trace variables are set on. If one or more of MPVM's trace flags are set on, MPVM writes tracing information to the virtual printer.

For example, suppose when a user typed in a certain string, EFGVIEW abnormally ended with an operation exception. You could follow these steps to trace the problem:

1. Use MPVM to establish a session with the system, and log on to your normal CMS user ID.
2. Spool your virtual printer to your reader with the following command:

   ```
   ?CP SPOOL E *
   ```

3. Get to the last known spot before the error occurred.

4. Issue the following command to turn on all trace flags:

   ```
   ?TRACES ON
   ```

   (Or you could turn individual trace flags on, using a command such as ?SET TRACE1 ON.)

5. Then issue the command that caused the error.
6. Exit MPVM's EFGVIEW module. (Issuing ?TEMPDISC or ?SESSTERM * accomplishes this.)

You will then get a trace file in your reader. If you explicitly closed your virtual printer, the trace file will have an identifier of *(none) (none)*; if you do not directly close the printer, the module does it automatically, and the resulting file is named MPVM TRACE. You can browse this file, and it will be useful for IBM personnel to track the problem.

Note: The trace output file could contain sensitive information, such as passwords. You may want to receive the file to your filemode A minidisk and edit any sensitive data before giving it to the problem determination personnel.

If you need to trace EFGVIEW early in its execution, before the TRACE variables are set, you can use a special parameter when issuing the MPVM command. For example, to turn on EFGVIEW tracing from the very start of a session, you could issue:

```
mpvm ?TRACES * mysys
```

TRACEn Subcommand

Use the TRACEn subcommand to control the tracing options for the EFGVIEW module, which runs on each MPVM user's virtual machine. For each trace option you set ON, MPVM will perform that type of trace. You can use the TRACE macro to turn all MPVM traces ON or OFF.

When you turn at least one trace option ON, the output (MPVM TRACE) is sent to your virtual printer when MPVM exits.

The TRACEn subcommand enables you to trace in these categories:

- Entry into and return from every routine
- Status information at various critical decision points
- All messages are logged
- Get_Buffer and Return_Buffer management
- IUCV parameter lists and data flow
- Console interrupts and commands
- Screen_Image routine.

Tracing the EFGVIEW Module

If you have your own interactive tracing tool, the EFGTRACE exec can also be used as a front end for tracing the EFGVIEW module.

As supplied, EFGTRACE simply invokes the specified module. However, if you edit EFGTRACE to include a call to your tracing tool, you can use EFGTRACE to trace the entire EFGVIEW module. Here are the steps to accomplish this:

1. Edit the EFGTRACE exec file to include a call your tool.
2. Issue the CMS command GLOBALV SELECT MPVM SET TRACE ON.

3. Issue the following from the CMS command line:

```
EFGTRACE EFGVIEW
```

When the MPVM command is invoked, and it sees that the CMS global variable TRACE variable is on, it invokes the EFGTRACE exec to trace the EFGVIEW module.

Logging Messages and Screen Images

You can control whether error messages and screen images are sent to output files.

The MPVM ERRLOG message file is placed on your filemode A minidisk.

You can control whether messages and screen images get written to error log files using the variables ERRLOG and ERRIMAGE. Note that you cannot get screen images logged without messages though.

Notes:

1. If messages are being logged to one of these files, and the disk becomes full, logging stops and message 272W is issued.
2. When logging display images, field attributes are included, but non-displayable fields are not included.

For identification purposes, MPVM messages are issued with a header that has this format:

EFG*abcnnns*

where:

EFG
is the product prefix for MPVM

abc
are three characters denoting where the message was issued from. If the message was issued from an MPVM macro, these are BSG. If the message was issued from an MPVM internal file, these are the fourth, fifth, and sixth characters of that file's name; see the next section for details.

nnn
is the message number

s is the severity code

Diagnosing PVMG Problems

PVM provides several ways to diagnose PVMG problems. This section describes methods of PVMG problem determination.

PVMG Trace Facility

The PVMG trace facility provides tracing for:

- System functions
- Line driver I/O.

Trace records created by the PVMG TRACE command are directed to internal PVMG wrap tables or to GTRACE tables, depending upon the type of trace specified, and trace output destination specified on the TRACE command.

Tracing PVMG System Activity

Use the TRACE command with the SYSTEM option to assist in problem source determination when internal PVMG trouble is expected. This command combination causes a chronological trace of the activities of key PVMG system components. The PVMG trace facility records requests for service, passed data (if any), execution of services, and status information.

The output resulting from the TRACE command can be directed to:

- An internal PVMG trace table
- A GTRACE table.

  If you use the GTRACE option, you should first enter the ETRACE command so that trace events are entered into the GTRACE table.

The TRACE command is described in "TRACE" on page 398.

PVMG I/O Tracing

Use the TRACE command with the Node *nodeid* and ON option to trace line driver activity for a specific *nodeid*. Tracing begins when line driver activity is started. Trace output is written to an internal PVMG trace I/O wrap table. (A PVMG server may have multiple I/O wrap tables.)

The TRACE command is described in "TRACE" on page 398.

Other PVMG Problem Determination Actions

PVMG will do basic error checking during its processing. If a problem such as:

- An unexpected post
- An error occurring when posting data to a target application
- A data buffer segment is received out of order

then the session is terminated.

For errors at the task level, PVMG will produce a dump of part or all of the PVMG virtual machine's storage, and PVMG terminates.

For errors regarding a user session, PVMG abnormally terminates the active task and specifies that a dump will be sent to the user's virtual reader. When information is dumped to the user's virtual reader, console messages are written.

The PVMG Dump Facility

If PVMG terminates abnormally, a dump will be produced. The dump will be a:

- CP dump
- VMDUMP (a VM IPCS dump).

Analyzing PVMG Dump Information

You can use the IPCS component of VM/ESA (370 Feature) and VM/SP, or the Dump Viewing Facility of VM/ESA and VM/XA to analyze PVMG system failures and then manage problem information and status. Using these facilities, you can interactively retrieve and examine information from PVMG storage dumps to help track and report PVMG problems. These facilities can also be used to examine the trace records in a CPTRAP reader file. For more information on creating a CPTRAP file in PVMG, see page 182.

IPCS and the Dump Viewing Facility also process dumps created by CP and CMS; however, this book gives only an overview of how these facilities operate with PVMG and tells how you can obtain information using IPCS commands. The Dump Viewing Facility supports obtaining information from PVMG dumps by supporting some IPCS commands. For more information about the IPCS operation and use, see the *VM/ESA: Interactive Problem Control System Guide and Reference for 370, SC24-5438*. For more information about the Dump Viewing Facility operation and use, see the *VM/ESA Dump Viewing Facility Operation Guide and Reference SC24-5530* or the *VM/XA System Product: Dump Viewing Facility Operation Guide and Reference, SC23-0359*.

DMTI end

Part 3. Modifying PVM

Chapter 13. Tuning PVM Performance 199
Enhancing PVM Performance 199
Installing MPVM in a Shared Segment 203
Developing Exits 209

Chapter 14. National Languages on Your PVM System 227
National Language Overview 227
Installing National Languages for PVM 2.1 228
Changing the Backup Language 235
Changing Message Destinations 236
Changing the PVM Translate Table 237
Creating and Using a Message Repository 238
Rules for Making Your Own Repository 240

Chapter 15. Facilities for Automating PVM and PVMG Operations 251
PVMPROP - A Programmable Operator Interface for PVM and PVMG 251
The Command Response Interface 252
PVMPROP/CRI Message Protocol 252
Using Command Prefixes 255
Sample Application 257

Chapter 16. Application-to-Application Overview 259
PCCF Data Flow 259
Application-to-Application Communication Sample Program 261

Chapter 17. Customizing CVIEW 265
Listed Consultants — CONSULT IDNAME 265
Program Function Key Assignments — DXGKEYS LIST 266
The Customizing EXEC — DXGINIT 267

© Copyright IBM Corp. 1991,1993

Chapter 13. Tuning PVM Performance

Enhancing PVM Performance

The PVM system programmer has to weigh the benefits of the various options discussed below against the effect on the overall system performance to determine which options should be chosen for the installation.

Certain CP options enhance PVM performance. One or more of these options can provide adequate performance on heavily loaded systems. PVM spends most of its time waiting for an interrupt to service. When an interrupt occurs, the virtual machine should be dispatched quickly for good user response time. See "Performance" on page 38 for VM/ESA and VM/XA considerations. For additional information, see *VM/ESA: CP Planning and Administration for 370, SC24-5430*.

SET QDROP OFF Option

This option eliminates the overhead in CP caused by scanning PVM page and segment tables. It also prevents resident pages from being put on the flush list whenever PVM enters an idle wait state.

The NOQ2 and NOQ3 options of the QDROP command improve the performance of a service machine like PVM. The NOQ2 option significantly reduces the erratic delays in PVM due to delays in Q2 and Q3.

The SET QDROP OFF option is not supported on VM/ESA or VM/XA systems.

SET FAVORED Option

This option prevents PVM from being dropped from the CP scheduler queue while in a wait state.

VCUNOSHR Directory Option

This option tells CP to create all virtual control units as unshared virtual control units, eliminating virtual control unit busy conditions. PVM is designed to work best with unshared channels and control units.

The VCUNOSHR option is not available on VM/ESA or VM/XA systems.

DIAGNOSE X'98' Real Channel Program Directory Option

The DIAGNOSE X'98' code allows a V=V user to lock I/O pages and create real CCWs and have CP start the I/O without doing I/O translation. For a virtual machine to use the diagnose, the DIAG98 option must be added to it's directory. PVM uses this diagnose code for the 3088 line driver.

The DIAG98 option is not available on VM/ESA and VM/XA systems.

© Copyright IBM Corp. 1991,1993

SET RESERVE Command

On systems with high paging load, PVM will very likely be paged out when an interrupt comes in. The SET RESERVE command, however, lets most PVM active pages remain in real storage.

The number of reserve pages can be estimated by sampling the working set of the running machine or can be roughly calculated by:

```
RES = 8 + 4(EMUL) + 3(BSCA) + 2(CTCA) + 7(R3270) + 5(APPC)
      + 2(IUCV) + 3(DIAL) + 2(PRT) + 2(PASS) + 2(APP)
      + 3(ROCF) + 2(PCCF) + 3(MPVM)
```

where:

EMUL
is 1 if any 327x line drivers are active.

BSCA
is 1 if any BSCA line drivers are active.

CTCA
is 1 if any CTCA line drivers are active.

R3270
is 1 if any R3270 line drivers are active.

APPC
is 1 if any APPC line drivers are active.

IUCV
is 1 if any IUCV line drivers are active.

DIAL
is 1 if any attached display support tasks are active.

PRT
is 1 if any attached printer support tasks are active.

PASS
is 1 if any users are using the PASSTHRU command.

APP
is 1 if any PVM users are accessing the local CP system.

ROCF
is 1 if any ROCF line drivers are active.

PCCF
is 1 if any PCCF sessions are active.

MPVM
is 1 if any users are using the MPVM command.

This algorithm provides a rough estimate. You may have to specify a larger value based on the individual activity and number of users.

Dispatching Priority

A priority can be specified on the USER control statement for PVM in the VM directory. CP uses the specified value, or the default value (64), to determine the PVM dispatching priority. The system operator can alter this value, for a given IPL, by using the CP SET command.

ROCF Sessions

The user may notice a slower response time during ROCF sessions when the VM system is heavily loaded. This is due primarily to the combination of the ROCF line speed (1200 bps) and the scheduling of PVM in a heavily loaded system. In light of this, the installation should:

- Review the performance options established for PVM to ensure their adequacy.
- Reserve use of the ROCF session for those interactions that require its exclusive capabilities, like remote IPL.

Tailoring the PVM Module

PVM consists of support routines which may never be used depending on the configuration of your installation. Parts of PVM that are not being used can have their support routines deleted from the build list (the PVM EXEC file for PVM 2.1 or PVMMOD EXEC for PVM 2.1.1). By removing parts of PVM in this way, you can:

- Improve PVM performance
- Reduce the virtual storage requirements for your PVM
- Simplify problem determination, if that becomes necessary.

DMTI

Diagnosis, Modification, and Tuning Information is provided *to help you optimize the performance* of *PVM and it's components*.

WARNING: Do not use this Diagnosis, Modification, and Tuning Information as a programming interface.

To remove a function of PVM, follow these steps:

1. Edit the PVM build list (PVM EXEC). To remove a function, place an asterisk (*) in the first column of the lines listing the object modules associated with the function. (You can later add the function back to PVM by replacing the asterisks with blanks.) The object modules associated with PVM functions are listed below.

Function	Description	Associated Modules
Remote 3270 emulation (327X) line driver	Emulates a remote 3270 bisynchronous control unit.	DVMSIM DVMSIT DVMSIU
BSC networking (BSCA) line driver	Allows PVM to communicate with another PVM virtual machine over a bisynchronous line.	DVMBSC DVMHSP
3088 channel-to-channel networking (3088) line driver	Allows PVM to communicate with another PVM virtual machine through an IBM 3088 Multisystem Communications Unit.	DVMC38 DVMC39 DVMC3A

Function	Description	Associated Modules
Channel-to-channel networking (CTCA) line driver	Allows PVM to communicate with another PVM virtual machine through a channel-to-channel adapter.	DVMCTC
Remote 3270 (R3270) line driver	Allows PVM to directly support remote 3270 bisynchronous control units.	DVMRMA DVMRMB DVMRMC DVMRMD DVMRME DVMRMF DVMRMG DVMRMH DVMRMI
APPC line driver	Allows PVM to directly communicate with other PVM nodes using APPC/VM protocol.	DVMAPA DVMAPB DVMAPC DVMAPD DVMAPE DVMAPF DVMAPG DVMAPH
IUCV line driver	Allows PVM to communicate with the local PVMG server	DVMVTA DVMVTB DVMVTC DVMVTD
Remote operator facility (ROCF) line driver	Lets you control a remote 4300 processor.	DVMRSF
Dialed display user interface	Allows users to access PVM by using the CP DIAL command.	DVMUGR
PASSTHRU user interface	Allows users to access PVM from their own virtual machine by using the PASSTHRU command.	DVMUIN
MPVM user interface	Allows users to access PVM from their own virtual machine and to have simultaneous active sessions.	DVMMPA DVMMPB DVMMPC DVMMPD DVMMPE DVMMPF DVMMPG EFGIMA EFGIMB
Attached printer support task	Allows attached 328X printers to be logically connected to remote 3270 emulator ports.	DVMUPR DVMUPS DVMUPT
Personal computer connect facility (PCCF)	Allows PVM to directly communicate with CMS applications using the inter-user communications vehicle (IUCV). Communication between VM applications and communication between VM to Personal Computer applications are supported.	DVMUIA DVMUIB DVMUIC DVMUID DVMUIE DVMUIF DVMUIG DVMUIH DVMUII DVMUIJ DVMUIK

2. Rebuild PVM 2.1 using the PVMBLD command. See page 315 for a description of the PVMBLD command. To rebuild PVM 2.1.1, see the Program Directory.

DMTI end

Installing MPVM in a Shared Segment

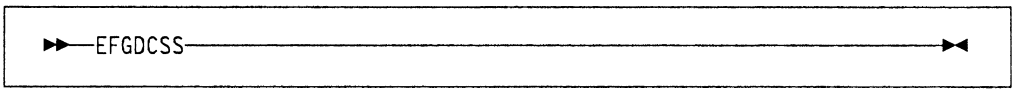
If your installation plans on running MPVM in a large number of virtual machines you may save on CP paging by installing MPVM in a shared segment.

By installing MPVM into a shared segment, you have one copy in shared storage, rather than individual copies in each user's virtual machine.

Purpose

The EFGDCSS exec, loaded to MAINT's 49E minidisk for PVM 2.1 or P684100E's 400 minidisk for PVM 2.1.1, will assist in the installation of a shared segment. The exec will prompt you for the necessary information required to generate a shared segment for your particular environment. The majority of installations can press the *enter* key in response to exec prompts thereby taking the default installation values for the shared segment generation method (SEGGEN, SAVESEG, or SAVESYS), segment location above or below 16 megabyte line, segment name, and segment page address. The syntax of the EFGDCSS exec follows.

Note: PVM 2.1.1 only uses the SEGGEN method of building shared segments. VM/ESA Release 2 customers using PVM 2.1.1 can use the VMSES/E support for building the MPVM shared segment. For further details on VMSES/E support, see the Program Directory.



Usage Notes

1. EFGDCSS must be issued from a user ID with the appropriate CP CLASS of CLASS E, or its equivalent.
2. The user ID issuing EFGDCSS must have a virtual machine size that includes the entire saved segment. For example, if a command is issued to load the MPVM segment at 5 megabytes, then the user ID that issues the command must have a virtual machine of at least 6 megabytes, since the MPVM shared segment requires .7 megabytes.
3. The user ID that issues the commands to save the shared segment above the 16 megabyte line must be in a non-370-mode virtual machine in order to address the storage above 16 megabytes.
4. If using the SEGGEN method of installing a shared segment, then a SYSTEM *segid* will be created on your A-disk which must be copied to the system 190 disk to complete the shared segment installation.
5. If your system is VM/SP or VM/ESA 370 based, then you will need to first define your MPVM shared segment in the DMKSNT ASSEMBLE file, rebuild the CP nucleus, and IPL the system before running the EFGDCSS exec. See "Sample DMKSNT Entries for VM/SP or VM/ESA 370 based Systems" on page 205 for more information about this subject.
6. Once the shared segment has successfully installed, the exec will issue a good completion message. You can then proceed to the "MPVM Shared Segment Verification" on page 204.

MPVM Shared Segment Verification

Once the EFGDCSS EXEC has been run, and the MPVM shared segment installation has completed, you may want to verify the shared segment using the following procedures.

1. Log onto a user ID on the same system as the shared segment with a virtual machine storage size less than or equal to the location of where the MPVM shared segment has been loaded.
2. Issue the MPVM command, the MPVM selection menu should appear.
3. Ensure that the shared segment is being used by using one of the following instructions based on your system level and generation method selected when you used the EFGDCSS exec.
   a. For VM/SP or VM/ESA 370 based systems issue

      ```
      ?NUCXMAP
      ```

      from the command line of the MPVM selection menu. If you are running with the MPVM shared segment, file EFGVIEW should not be among the resulting list of file names.

   b. For VM/XA or VM/ESA systems, logon to a second user ID with class "E" privilege or equivalent, and issue the following command:

      ```
      Q NSS USERS segname
      ```

      Where *segname* is the file name of the PSEG file used for the SEGGEN shared segment generation method, or the shared segment entry name for the SAVESEG or SAVESYS shared segment generation method (depending upon which generation method was used when the EFGDCSS exec was executed).

      The output will contain any user IDs currently using the MPVM shared segment.

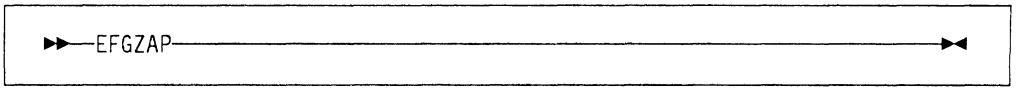
Changing the Names of MPVM Shared Segments

The names of MPVM shared segments are hardcoded in the EFGVSTUB MODULE. They are:

- MPVM, for the saved segment less than or equal to 16 megabytes
- MPVMHIGH, for the saved segment above 16 megabytes.

Purpose

You can change this hardcoding to support your own names for the low or high segments by using the EFGZAP exec. This allows an installation to have multiple (and different) copies of the MPVM shared segment. This capability is useful when testing new MPVM shared segments before they are placed in production. The syntax of the EFGZAP exec follows.



Usage Notes

Before running the EFGZAP exec, verify you have a copy of the EFGVSTUB MODULE available on a minidisk that is in R/W mode so that the EFGVSTUB MODULE can be updated.

Sample DMKSNT Entries for VM/SP or VM/ESA 370 based Systems

Two sample DMKSNT ASSEMBLE entry files are provided on MAINT's 191 minidisk for PVM 2.1 and one sample DMKSNT ASSEMBLE entry file is provided on P684100E's 2C2 for PVM 2.1.1 minidisk to assist you with defining an MPVM shared segment on a VM/SP or VM/ESA 370 based systems:

1. MPVM SNTENTRY which contains sample entries MPVM and MPVMDISK. MPVM can be used by the EFGDCSS exec for users that are using either the SAVESEG of SAVESYS method of shared segment installation. MPVMDISK can be used to contain MAINT's 38F minidisk (MPVM common disk) master file directory.
2. MPVMP SNTENTRY which contains a sample MPVM shared segment for VM/SP 6 or VM/ESA 370 sytems that will be using the EFGDCSS execs default method of SEGGEN for generating an MPVM shared segment.

These samples are similar, but MPVMP SNTENTRY is slightly larger, to accomodate the MPVM execs and macros.

Adding Common MPVM Commands and Macros to Your Installation Shared Segment

You can load your installation shared segment (default name is CMSINST) with the most commonly used MPVM commands and macros to improve response time.

Note: If you have not used the SEGGEN method of building shared segments, do the steps that follow.

1. Issue the following command to build CMSINST as a physical shared segment:

   ```
   DCSSGEN  fn ft fm  CMSINST
   ```

   where *fn ft fm* identifies a file that lists all the commands and macros that you want to load into the saved segment. A file named CMSINST MPVMLIST, supplied on MAINT's 49E minidisk, contains a sample list of these files; if you use that file, your command would be:

   ```
   DCSSGEN  CMSINST MPVMLIST fm CMSINST
   ```

   The DCSSGEN command—available on MAINT's 193 disk—requires that the executing user ID have authority to issue SAVESYS (class E or equivalent).
2. After loading these often-used macros and commands into the CMSINST installation shared segment, you must ensure that the CMSINST copies of the macros are found before the versions on MAINT's 38F minidisk. Update your VM system profile to include the following line:

   ```
   SET INSTSEG ON fm
   ```

   where *fm* is a file mode that is accessed ahead of MAINTs 38F minidisk.

If the CMSINST segment is changed, you can set up MPVM to automatically warn users that they must re-IPL CMS to pick up the latest changes.

To accomplish this, the person doing the DCSSGEN must do these steps:

1. Edit the sample file $CMSINST EXEC on the MPVM common disk (38F) and increment the *level = nn* in that file.
2. Save the $CMSINST EXEC file along with the others in the CMSINST segment. (The name $CMSINST EXEC gets added to the CMSINST MPVMLIST used for the DCSSGEN.)

The supplied sample SYSIPROF MPVM macro contains provisions to automatically warn MPVM users, if necessary, to re-IPL to get up-to-date information. For more information on the CMS installation shared segment, see the *VM/ESA: Installation for 370, SC24-5431* book.

Saving MPVM Common Disk Information in a Segment

You can place file directory information for the MPVM common disk (by default, the PVM user ID's 199 disk) into a shared segment. All MPVM users who link and access the common disk then share this same *master file directory*, providing faster disk access time and reduced storage usage.

To place this information in a shared segment, do these steps:

1. Get logged on to the MAINT user ID.
2. Create a discontiguous shared segment (DCSS), one segment in size, called *MPVMDISK* to contain the MPVM Common Disk master file directory.

   For VM/SP Release 5 and VM/SP HPO Release 5, a file named MPVM SNTENTRY, supplied on MAINT's 191 minidisk, includes an example skeleton entry you can use as a model. For all later VM versions, you must use the DEFSEG command.
3. Link to MPVM common disk in R/W mode.

   Note: The MPVM common disk is MAINT's 38F disk for PVM 2.1 and P684100E's 403 disk for PVM 2.1.1.
4. Issue the following command to write the segment name on the disk label record:

   ```
   SAVEFD INIT linkaddr label segname
   ```

   where *linkaddr* is the address you used in your link in step 3, *label* is the CMS minidisk label of MPVM common disk and *segname* is the name of the MPVM shared segment (as in MPVMDISK).

   Note: The SAVEFD command requires class E authority (or equivalent).
5. Issue the following command to save file status tables on the segment:

   ```
   SAVEFD SAVE linkaddr label segname
   ```

   where *linkaddr* is the address you used in your link in step 3, *label* is the CMS minidisk label of MPVM Common disk, and *segname* is the name of the MPVM shared segment (as in MPVMDISK).

Note: The master file directory must be resaved (by linking to the disk and then issuing SAVEFD SAVE) any time a file is changed on the disk, or any time the disk is accessed in R/W mode and then released.

For more information on the SAVEFD command and defining segments, see the *VM/ESA: CP Planning and Administration* and the *VM/ESA: CP Planning and Administration for 370* books.

Fine-Tuning the PVM Server

You can use some privileged CP commands to tune your PVM server virtual machine:

- Issuing CP's SET PRIORITY command, which specifies the processor's dispatching priority in relation to other virtual machines.

  The format of the command is *SET PRIORITY userid level*. The priority *level* to supply depends on your particular system, but a value of 50 is probably a good start. (A priority of 50 is higher than what an *average* system user gets, 64.)

  Note: Be sure not to give the PVM server a higher priority than the GCS and VTAM user IDs have.

- Issuing the SET FAVORED command. This privileged CP command specifies the percentage of processor time that is used for the PVM server.

  The format of the command is *SET FAVORed userid percentage*. The *percentage* value to supply depends on your particular system.

- Issuing the SET QDROP command. This privileged CP command eliminates some paging overhead on virtual machines that frequently communicate with other virtual machines.

  The format of the command is *SET QDROP userid OFF*.

- For more information on the SET PRIORITY, SET FAVORED, and SET QDROP commands, see the *VM/ESA: CP System Command Reference for 370, SC24-5434*.

Checking Response Time for User Commands

The RTM macro, which is part of the MPVM user interface, can records response times for every MPVM user interaction with a target application. For more information on the RTM macro, see the *VM/Pass-Through Facility Programming Reference*.

Customizing MPVM Sessions

PSPI

CAUTION

Only an experienced system programmer should attempt to write exit routines for PVM. The writing of an exit routine requires a thorough knowledge of system programming, of PVM programming conventions, and of PVM internals. If, without having this knowledge, you attempt to write exit routines or install new exit points, you run the risk both of seriously degrading the performance of your system and causing system failure.

Customizing PVM, therefore, is intended to be accomplished primarily by experienced system programmers.

Also, be aware that exit routines involving user display stations, PVM, or target sessions could affect extended data streams. Any alteration to extended data streams can produce errors. Programmers should be aware if data streams are involved. Test your exit routines thoroughly.

MPVM provides several ways to set up sessions to meet your needs. You can:

- Write your own automatic signon exec
- Write execs and macros, which get called by MPVM during its session processing, to tailor MPVM to specific user needs
- Supply your own meaning for reserved MPVM variable slots
- Write macros that get triggered by certain application conditions, or by the pressing of *hot keys*.

These capabilities are described in the *VM/Pass-Through Facility Programming Reference*.

Sample MPVM Exits

Currently, MPVM puts out all messages as fullscreen 3270 erase write datastreams. This exit will put each message in linemode via the REXX SAY command.

```
/* Sample exit9 macro with filename of exit9 and filetype of mpvm */
trace 0
if mpvmvalu('msgmode') then
  do
  say mpvmvalue('lastmsg')
  'set lastmsg'
  end
exit 0
```

Figure 33. Sample exit for issuing messages

A sample exit1 macro to cause an MPVM session to terminate with the '####' string (like passthru)

```
/*Sample exit1 macro with filename of exit1 and filetype of mpvm*/
trace 0
'EXTRACT /LASTINP/'
if length(mpvm_lastinp.1)  = 10     &,
   right(mpvm_lastinp.1,4) = '####'
   then  'SESSTERM'
exit 0
```

Figure 34. Sample exit for using ####

To have these exits automatically set when ever MPVM is used, create an MPVM macro with filename of startup as shown in the following example:

```
/*exit9 is called just before MPVM issues a message to the terminal*/
/*exit1 is called anytime the keyboard input is read from the terminal*/
'set exit9 exit9'
'set exit1 exit1'
```

Figure 35. Sample STARTUP MPVM file

Developing Exits

A system programmer can make changes to PVM by developing exits. There are three areas of PVM processing that you can customize with exits. These areas, and the PVM modules associated with them, are:

Session control exits — These exits can make use of external reference points for session initiation or termination and for command and command response processing. The session control exits are associated with module DVMNUE.

Networking management exits — Exits applied here can be used to collect data about a user's session. The networking management exits are associated with module DVMNME.

CMS user exits — These exits provide a means for collecting data on a user's I/O activity. The CMS user exits are associated with module DVMCUE, which is part of the DVMUSI module.

Exit Point and Module Summary

The following table provides a reference to the IBM-defined exit points and the PVM and DVMUSI modules from which they are called. The MPVM component allows you to develop exits. For information about developing exits for MPVM, see the *VM/Pass-Through Facility Programming Reference*.

Exit Point	Name	Module
DVMNUE1	Session Initiation Exit	DVMNUE
DVMNUE2	Session Termination Exit	DVMNUE
DVMNUE3	Origin Node Session Request Exit	DVMNUE
DVMNUE4	Intermediate Node Session Request Exit	DVMNUE
DVMNUE5	Command and Response Exit	DVMNUE
DVMNUE6	CINIT Session Initiation Data Exit	DVMNUE
DVMNMEC	Cluster Statistics Exit	DVMNME
DVMNMEL	Link Statistics Exit	DVMNME
DVMNMES	Session Statistics Exit	DVMNME
DVMCUEBI	Before Initialization Exit	DVMCUE
DVMCUEAI	After Initialization Exit	DVMCUE
DVMCUEBR	Before Read Exit	DVMCUE
DVMCUEAR	After Read Exit	DVMCUE
DVMCUEBW	Before Write Exit	DVMCUE
DVMCUEAW	After Write Exit	DVMCUE
DVMCUEBF	Before Final Exit	DVMCUE

Exit routines are never required as part of standard PVM processing. A module supplied by IBM supports these areas with dummy exit routines that immediately return control to the caller. If you decide to provide additional processing for any of these exits, you must replace and assemble the source code for the corresponding routines. Also, you must either rebuild the:

- PVM load module using the PVMBLD command (for the session control and the networking management exits)
- DVMUSI module that executes in the user's CMS virtual machine using the PVMGEN command (for the CMS user exits).

Note: For PVM 2.1.1, see the Program Directory for applying local modifications. See "Rebuilding PVM for Exits" on page 225 on how to apply your exit routines to your installation.

See the Appendix J, "PVM Access Security Exits" on page 439 for reference information about the supplied sample PVM access security exits.

Programming Note

Registers 10 — 13 contain the following information for session and management control exits:

- Register 10 is used as the base addressing register
- Register 11 contains the address of the system control block (DVMSYSB)
- Register 12 contains the address of the session manager's (DVMNET's) task control block (DVMTCB)
- Register 13 contains a pointer to the register stack area.

Registers 11 and 12 (and the data areas they point to) must not be modified. Registers 10 and 13 may be modified (using only the DVMENTR macro), but must be restored (using the DVMRETN macro).

Any other registers used (including 10 and 13) must be restored to their original values.

To save working registers at entry use the DVMENTR macro. To restore the registers upon exiting, use the DVMRETN macro. The sample exit shown in Figure 36 on page 218 illustrates the use of these macros.

For MPVM users, messages pointed to by R1 are truncated to 35 bytes upon return from the session control exits, due to an internal PVM restriction.

Some of the exits discussed in this section may have one or more of the following control blocks addressed by an input register. These control blocks provide data that the exits may use, but the following control blocks must not be altered:

- DVMACT see "DVMACT (Cluster Table Entry) Format" on page 420
- DVMALT see "DVMALT (Active Link Table Entry) Format" on page 420
- DVMAUT see "DVMAUT (Active User Interface Link Descriptor) Format" on page 423
- DVMBUF see "DVMBUF (Internal Buffer Header)" on page 425
- DVMPORT see "DVMPORT (Simulator Link Port Table Entry)" on page 430
- DVMSYSB see "DVMSYSB (PVM Multitasker System Control Block) Format" on page 431
- DVMTCB for session manager see "DVMTCB" on page 434

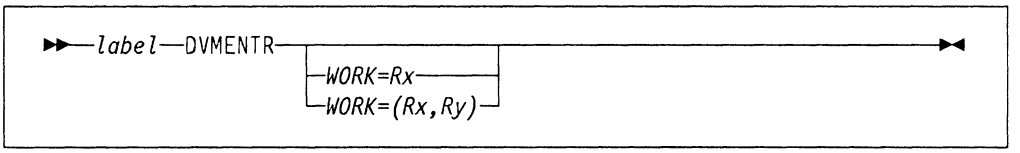
PVM Macros Used in the Sample Access Security Exits

Three PVM macros are used by the sample access security exits. This section contains a brief description of how they are used.

DVMENTR

DVMENTR is used to generate the standard entry protocol for subroutines in PVM. It saves various registers on a register stack for later restoration by the DVMRETN macro.

Format



Operands

label
the routine entry name

WORK=Rx
WORK is the macro variable. Rx is the register that you want saved.

WORK=(Rx,Ry)
WORK is the macro variable. Rx,Ry is the register range that you want saved.

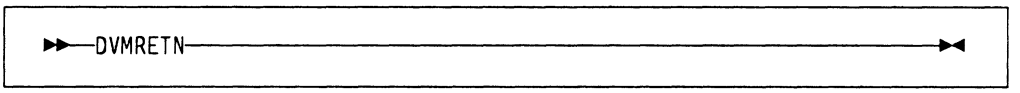
Usage Notes

1. If the WORK keyword is included, the register or register range specified will be saved on the stack and restored by the DVMRETN macro.
2. Registers always saved by the macro:
   - R10 (used as a base addressing register)
   - R14 (return address register)
   - R13 is *bumped* by four times the number of registers saved including R10, R14, and the number specified on the WORK option.

DVMRETN

DVMRETN is used to generate the standard return protocol for subroutines in PVM. It restores the registers saved by the DVMENTR macro, (including R10, R13, and R14, and any specified on the WORK option of the DVMENTR macro), then returns to the subroutine's caller using the address specified in R14.

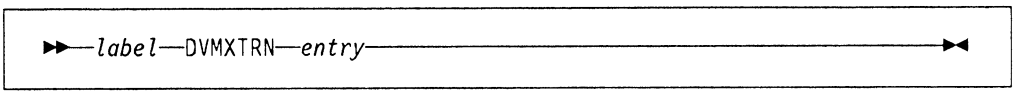
Format



DVMXTRN

DVMXTRN is used to map internal subroutine names into external linkedit names.

Format



Operands

label
is the internal subroutine name.

entry
is the external linkedit name.

Session Control Exits

The DVMNUE module provides for six session control exit routines. These routines are described in the following sections.

Session Initiation Exit (DVMNUE1)

Session control exit, DVMNUE1, is provided upon initiation of a PVM session.

This routine is invoked at session initiation to allow tracking sessions in which the target node is the:

- Local node
- Emulator or ROCF nodes defined in the local node's configuration file.

The exit routine may be programmed to allow session initiation to proceed or not. For emulator sessions, it can also provide a different port for the session than was requested.

Note that execution of either session control exit DVMNUE3 (local requests) or DVMNUE4 (nonlocal requests) precedes execution of this exit.

Register Contents on Entry to the User Routine

Register	Contents
R1	Address of a 39-byte area to provide an explanatory message if the session is to be aborted
R2	Address of a parameter list containing: **Byte** **Contents** **00-07** Originating node ID **08-15** Originating user ID **16-19** Address of DVMPORT control block list (hexadecimal), if accessing a port on an emulator link or zero, if not **20-21** Port number (binary). Valid if bytes 16-19 are not zero. **22-23** User display station address (hexadecimal) **24-25** User display station class and type **26** User display station device model (hexadecimal) **27-34** Destination node ID **35-42** EBCDIC real terminal address **43-50** Origin node if this is a R3270 session, origin *vmid* if this is an MPVM session or PCCF session **51-58** Destination *userid*

Register	Contents
R3	Address of DVMALT control block for emulator line if connection is to a port on an emulator link.
R11	Address of the system control block (DVMSYSB). Do not modify.
R12	Address of session manager's (DVMNET's) task control block (DVMTCB). Do not modify.
R13	Address of register save stack area
R14	Return address
R15	Entry address.

You can access the listed control blocks by using the PVM macro library (DVMMAC MACLIB).

Register Contents on Return to PVM

Register	Contents
R3-R9	Save on entry using the DVMENTR macro; restore on exit using the DVMRETN macro.
R10-13	R11 and R12 must not be modified. R10 and R13 must be restored to their original value.
R15	Must contain one of the following return codes: **0** Tells PVM to continue the session. **4** For emulator links only, this means try another port. For non-emulator links, it tells PVM to terminate the session. **8** Tells PVM to terminate the session. If R15 does not contain zero on return, an explanatory message must be returned in the 39-byte area (pointed to by R1 on entry to the exit routine) which is displayed on the user's screen.

Session Termination Exit (DVMNUE2)

The second session control exit to optional user routines, DVMNUE2, is provided at session termination.

This routine is invoked at session termination to allow tracking the termination of sessions in which the target node is:

- The local node
- Emulator or ROCF nodes defined in the local node's configuration file.

Register Contents on Entry to the User Routine: Register usage and the parameter list passed are very similar to that described above for session initiation. The only difference is that R1 does not contain the address of an area for returning a reason the session cannot continue. Since the session is now completed, the area is not needed.

Register Contents on Return to PVM: Register usage and the parameter list return to PVM is also very similar to that described above for session initiation. The only difference is that for the session termination exit, zero must be returned in R15.

Session Request Exits (DVMNUE3 and DVMNUE4)

The third and forth session control exits, DVMNUE3 and DVMNUE4, are provided for session requests originating at the local node and for session requests originating outside the local node. They occur when the node receives the session request.

The Origin Node Session Request Exit (DVMNUE3) routine is invoked at session initiation to allow tracking of any or all session requests originating from the local node. The exit may be programmed to allow the request to proceed or not. It can also alter the requested port and/or destination.

The Intermediate Node Session Request Exit (DVMNUE4) routine is invoked at session initiation to allow tracking of session requests originating from another node. The exit may be programmed to allow the request to proceed or not. It may also alter the requested port and/or destination.

Register Contents on Entry to the User Routine

Register	Contents
R1	Address of a 39-byte area to provide a reason if the session is to be aborted
R2	Address of a parameter list containing: **Byte** **Content** **00-07** Originating node ID **08-15** Originating user ID **16-19** Unused **20-21** Port number (binary) or zero **22-23** User display station address (hexadecimal) **24-25** User display station device class and type **26** User display station device model (hexadecimal) **27-34** Destination node ID **35-42** EBCDIC real terminal address **43-50** Origin node if this is a R3270 session, origin *vmid* if this is an MPVM or PCCF session **51-58** Destination *userid*
R11	Address of the system control block (DVMSYSB). Do not modify.
R12	Address of session manager's (DVMNET's) task control block (DVMTCB). Do not modify.
R13	Address of the register save stack area
R14	Return address
R15	Entry address.

You can access the listed control blocks by using the PVM macro library (DVMMAC MACLIB).

Register Contents on Return to PVM

Register	Contents
R3-R9	Save on entry using the DVMENTR macro; restore on exit using the DVMRETN macro.
R10-13	R11 and R12 must not be modified. R10 and R13 must be restored to their original value.

Register	Contents
R15	Must contain one of the following return codes: **0** Tells PVM to allow the session to continue. **4** Tells PVM to try another destination (because the destination and/or port has been altered by the installation within the exit routine. If neither field is altered, the session is terminated.) **Note**: The capability to alter node names allows for use of synonym node names. An example is an installation which has two MVS systems running CICS accessible through PVM. The installation wants the ability to change which MVS system the users actually access for a specific CICS application. The installation can setup a synonym node name for the application, for example, CICSAPP1. A session control exit can be designed to recognize the request for *node* CICSAPP1, choose which real node the user is to access, and then alter the parameter list to point to the new (real) node as the destination. **8** Tells PVM to terminate the session. If R15 does not contain zero on return, an explanatory message must be returned in the 39-byte area (pointed to by R1 on entry to the exit routine) which is displayed on the user's screen.

Command and Response Exit (DVMNUE5)

The fifth session control exit, DVMNUE5, is provided before PVM processes any commands or command responses originating at another node.

This routine is invoked before PVM processes any commands or command responses originating at another node. The exit may be programmed to allow the command or response to continue or not. It may also change the destination of the command or response. If the target node is the origin node, this exit is not entered.

Register Contents on Entry to the User Routine

Register	Contents
R1	Address of a 39-byte area to provide a reason if the session is to be aborted.
R2	Address of a parameter list containing: **Byte** **Content** **00-07** Originating node ID **08-15** Originating user ID **16-19** Address of the buffer containing the command or command response **20-26** Unused **27-34** Destination node ID.
R3	Address of DVMALT control block for the outgoing node. If the value in R3 is not positive, then the destination node is unknown. The link referenced by R3 may not actually be available. The condition code from the route lookup routine is in the high byte of R6. An *SPM R6* instruction restores the condition code. A CC=0 indicates the path is available. A CC=1 indicates the node is unknown, and a CC=2 indicates the path is not connected. Do not modify.
R4	Address of the DVMALT control block for incoming link
R5	Address of the message packet (DVMBUF). Do not modify.
R11	Address of the system control block (DVMSYSB). Do not modify.
R12	Address of session manager's (DVMNET's) task control block (DVMTCB). Do not modify.
R13	Address of the register save stack area
R14	Return address
R15	Entry address.

You can access the listed control blocks by using the PVM macro library (DVMMAC MACLIB).

Buflen is a halfword in the Internal Buffer Header (DVMBUF) data area containing the length of the command. Bufdata contains the start of the command. Check the BUFSPEC field in the Internal Buffer Header before you decide to reject the message. If the reply bit in BUFSPEC is on, do not reject the message. For further information on DVMBUF, see "DVMBUF (Internal Buffer Header)" on page 425.

Register Contents on Return to PVM

Register	Contents
R3-R9	Save on entry using the DVMENTR macro; restore on exit using the DVMRETN macro.
R10-13	R11 and R12 must not be modified. R10 and R13 must be restored to their original value.
R15	Must contain one of the following return codes: **0** Tells PVM to process the command or command response normally. **4** Tells PVM to try another destination (because the destination has been altered by the installation within the exit routine.) **8** Tells PVM to return the reason code as a response. If R15 does not contain zero on return, an explanatory message must be returned in the 39-byte area (pointed to by R1 on entry to the exit routine) which is returned as a response to the original command.

CINIT Session Initiation Data Exit (DVMNUE6)

The sixth session control exit, DVMNUE6, is provided for examining the CINIT data that will be passed from an MPVM user for a session request to an SNA application that requires CINIT data.

This exit is provided for examining CINIT data that will be passed for a session control request to an SNA application that requires CINIT data. The routine will be invoked at session initiation time by the local node. The CINIT data can be altered and added to by the exit, but cannot exceed a length of 255 bytes. The exit can be programmed to allow the request to proceed or not proceed.

Register Contents on Entry to the CINIT Exit

Register	Contents
R1	Address of a 39-byte area to provide a reason if the session is to be aborted.
R2	Address of a parameter list containing: **Byte** **Content** **00-07** Originating node ID **08-15** Originating session ID (in form of *SESnnnn) **16-19** Unused **20-21** Port number (binary) or zero **22-23** User display station address (hexadecimal) **24-25** User display station device class and type **26** User display station device model (hexidecimal) **27-35** Destination node ID **35-42** EBCDIC real terminal address **43-50** MPVM user real VM ID
R3	Address of start of CINIT data
R4	Length of CINIT data
R11	Address of the system control block (DVMSYSB). Do not modify.

Register	Contents
R12	Address of session manager's (DVMNET's) task control block (DVMTCB). Do not modify.
R13	Address of the register save stack area
R14	Return address
R15	Entry address.

You can access the listed control blocks by using the PVM macro library (DVMMAC MACLIB).

Register Contents on Return to PVM

Register	Contents
R0-R3, R5-R9	Save on entry using the DVMENTR macro; restore on exit using the DVMRETN macro.
R4	The original or updated CINIT length
R10-13	R11 and R12 must not be modified. R10 and R13 must be restored to their original value.
R15	Must contain one of the following return codes: **0** Tells PVM to allow the session to continue **4** Tells PVM to try another destination (because the destination has been altered by the installation within the exit routine. If the destination has not been altered, the session is terminated.) The capability to alter node names allows for use of synonym node names. An example is an installation which has two MVS systems running CICS accessible through PVM. The installation wants the ability to change which MVS system users actually access for a specific CICS application. The installation can setup a synonym node name for the application, for example, CICSAPP1. A session control exit can be designed to recognize the request for "node" CICSAPP1, choose which real node the user is to access, and then alter the parameter list to point to the new (real) node as the destination. **8** Tells PVM to terminate the session If R15 does not contain zero on return, an explanatory message must be returned in the 39-byte area (pointed to by R1 on entry to the exit routine) which is returned as a response to the original command.

Sample Session Control Exit

An example of a session control exit follows. This code is added (by editing) to the dummy exit routine DVMNUE in the PVM at RMTPVM4. This routine checks to see if the remote user is attempting to connect to emulator port number 1 on node MVSAPL. If so, the origin node name is checked. If the name begins with *RMTPVM*, the session is allowed to complete. If not, a code of 4 is returned.

```
NUE1EXIT DVMENTR WORK=(R3,R9)
         PUSH  USING
         USING DVMEXIT,R2
         USING DVMALT,R3
*  If there is no port pointer, it is not an emulator line
         ICM   R15,15,UCALPADR
         BZ    ALLOW                NOT AN EMULATOR LINE
*  We can now check the nodename
         CLC   ALTNODE,=CL8'MVSAPL'  IS THIS FOR MVSAPL?
         BNE   ALLOW                NO...NO CHECK NEEDED
         LH    R15,UCALPORT         GET PORT NUMBER
         CL    R15,=F'1'            IS THIS PORT NUMBER 1?
         BNE   ALLOW                NO...ALLOW CONNECTION
         CLC   =C'RMTPVM',UCALORIG  IS THE USER AT A NODE
*                                   STARTING WITH 'RMTPVM'?
         BE    ALLOW                YES...ALLOW IT
         MVC   0(L'MSG1,R1),MSG1    INSERT MSG
         B     REJECT               GO REJECT IT
ALLOW    EQU   *
         SR    R15,R15              SET RETURN CODE
         DVMRETN ,           AND RETURN
REJECT   EQU   *
         LA    R15,4                SET RETURN CODE
         DVMRETN ,           AND RETURN
         EJECT
MSG1     DC    CL39'PORT 1 RESERVED FOR RMTPVM NODES'
DVMEXIT  DSECT
UCALORIG DS    CL8           ORIGIN NODE
UCALOUID DS    CL8           ORIGIN USERID
UCALPADR DS    A             PORT SLOT ADDRESS
UCALPORT DS    XL2           PORT NUMBER
UCALREAL DS    XL2           DEVICE ADDRESS
UCALCLAS DS    XL2           DEVICE CLASS AND TYPE
UCALMDL  DS    X             DEVICE MODEL NUMBER
UCALDEST DS    CL8           DESTINATION NODE
UCALRADR DS    CL8           EBCDIC REAL TERMINAL ADDRESS
UCALOVMI DS    CL8           ORIGIN NODE IF R3270
         EJECT
         COPY  DVMBUF
         COPY  DVMALT
         DVMTCB
         DVMSYSB
         REGEQU
```

Figure 36. Sample session control exit

Network Management Exits

The DVMNME module provides for three network management exit routines:

Cluster Statistics Exit (DVMNMEC) — This routine is invoked every time a R3270 cluster enters one of three states:

UP	PVM has been connected to the cluster.
IN-ERR	PVM has detected an I/O error on a cluster and is taking error recovery action.
DOWN	The cluster is being terminated.

The exit routine records the statistics in either a disk file, a spool file, or through the CP ACCOUNT facility.

Link Statistics Exit (DVMNMEL) — This routine is called at:

- PVM startup
- Link startup
- Link connect
- Link disconnect
- Link deactivation

- On a periodic basis for a connected link.

The exit routine records the statistics in either a disk file, a spool file, or through the CP ACCOUNT facility.

Session Statistics Exit (DVMNMES) — This routine records session statistics at session termination. The exit is **not** called when the input byte count passed to the DVMNMEL exit is zero. The routine is entered at each node along the session path.

The exit routine records the statistics in either a disk file, a spool file, or through the CP ACCOUNT facility.

Cluster Statistics Exit (DVMNMEC)

This exit, DVMNMEC, receives control when a R3270 cluster enters one of the three available states (UP, IN-ERR, and DOWN).

Register Contents on Entry to the User Routine

Register	Contents
R0	Contains a value specifying the state of the cluster. **0** The cluster is now active (UP). Sense bit information is not available at this entry. **1** The cluster is in the error recovery state (IN-ERR). **2** The cluster has been dropped from regular service (DOWN). The cluster is checked every 5 minutes to see if it is back in service.
R1	Address of a parameter list containing: **00-03** Address of the DVMALT control block for the link **04-05** Virtual address of the line (hex) **06-07** Real address of the line (hex) **08-11** Address of the DVMACT control block for the cluster **12-13** Cluster number (binary). The following field is available only if R0 is 1 or 2: **14-29** Sense bit counts (eight half-word binary numbers).
R11	Address of the DVMSYSB control block. Do not modify.
R12	The task control block of the running task. Do not modify.
R14	Return address
R15	Entry address.

You can access the listed control blocks by using the PVM macro library (DVMMAC MACLIB).

Register Contents on Return to PVM

Register	Contents
R10-R13	R11 and R12 must not be modified. R10 and R13 must be restored to their original value.

Link Statistics Exit (DVMNMEL)

The second network statistics exit, DVMNMEL, is provided at:

- PVM startup
- Link startup
- Link connect
- Link disconnect
- Link deactivation

- On a periodic basis for a connected link.

Register Contents on Entry to the User Routine

Register	Contents
R0	Contains a value that specifies when this exit was called. **0** Indicates this is the initial startup of PVM. In this case, there is no data in the parameter list. **1** Indicates a link startup. At this point, PVM has processed the START LINE command and is ready to attach a line driver task, and the line has not entered the connect state. Only the first three fields of the parameter list are valid at this call. **2** The link has been connected. At this point, the link has gone from the active state to the connect state. Only the first three fields of the parameter list are valid at this call. **3** The link has gone from the connect state to the active state. **4** The link has gone from the active state to the inactive or down state. Only the first three fields of the parameter list are valid at this call. **5** The link is in the connect state. This entry is invoked on a periodic basis set by the STATTIME record in the configuration file.
R1	Address of a parameter list containing: **00-03** Address of the DVMALT control block for the link **04-05** Virtual address of the line (hex) **06-07** Real address of the line (hex). The following fields are available only if R0 is 3 or 5: **08-11** Input block count **12-15** Output block count **16-19** Input byte count **20-23** Output byte count **24-25** Extra records per block on input **26-27** Extra records per block on output **28-29** Link delay time in milliseconds (network links only).
R3	Address of the DVMALT control block. Do not modify.
R11	Address of the DVMSYSB control block. Do not modify.
R12	The task control block of the running task. Do not modify.
R14	Return address
R15	Entry address.

You can access the listed control blocks by using the PVM macro library (DVMMAC MACLIB).

The *extra records* fields indicate the degree of blocking on the link. If a transmission block contains four message packets, then the block count is incremented by one, and the extra records field is incremented by 3. Only network links have the capability of blocking records.

The *link delay time* field indicates how long it took a special time stamped record to be sent down the link and returned by the other side. This field is available only on network links.

Register Contents on Return to PVM

Register	Contents
R10-R13	R11 and R12 must not be modified. R10 and R13 must be restored to their original value.

Session Statistics Exit (DVMNMES)

The session statistics exit, DVMNMES, records session statistics at session termination. This exit is not called if the input byte count from the Link Statistics Exit is zero.

Register Contents on Entry to the User Routine

Register	Contents
R1	Address of a parameter list containing: **00-07** Originating node ID **08-15** Originating user ID **16-23** Destination node **24-31** Destination user ID **32-35** Session start time (top 32 bits of TOD clock) **36-39** Input byte count **40-43** Output byte count **44-45** Origin device real address **46-49** Real device class, type, model, and features **50-57** Origin node ID if this is a R3270 session, origin VMID if it's an IUCV session, or blank **58-65** Destination VMID if this is an IUCV session, or blank.
R11	Address of the DVMSYSB control block. Do not modify.
R12	The task control block of the running task. Do not modify.
R14	Return address
R15	Entry address.

You can access the listed control blocks by using the PVM macro library (DVMMAC MACLIB).

Register Contents on Return to PVM

Register	Contents
R10-R13	R11 and R12 must not be modified. R10 and R13 must be restored to their original value.

CMS User Exits

There are seven user exits in the DVMCUE module of DVMUSI. These exits support a CMS user virtual machine participating in a PVM PASSTHRU session. The CMS User exits are:

- Before Initialization (DVMCUEBI)
- After Initialization (DVMCUEAI)
- Before Read (DVMCUEBR)
- After Read (DVMCUEAR)
- Before Write (DVMCUEBW)
- After Write (DVMCUEAW)
- Before Final (DVMCUEBF).

For each exit routine:

- R0-12 may be altered and need not be restored.
- R13 must be restored to its original value.
- R15 is used for linkage. You must use R15 for exit return code. Clear R15 for a normal return.

On return, R0 and R1 have significance for some routines.

Before Initialization Exit (DVMCUEBI)

This exit, DVMCUEBI, is provided to allow for any user initialization activity.

Register Contents on Entry to the User Routine

Register	Contents
R14	Return address
R15	Entry address.

Register Contents on Return to PVM

Register	Contents
R13	Must be restored to its original value.

This exit does not process return codes.

After Initialization Exit (DVMCUEAI)

This exit, DVMCUEAI, is scheduled after the PVM CMS application module, DVMUSI, has completed its initialization. The PASSTHRU command parameters have been processed and validated. These parameters cannot be changed by this exit routine. A session has not yet been established with the PVM system.

This exit is always scheduled prior to any of the read and write exits.

Register Contents on Entry to the User Routine

Register	Contents
R1	Address of a parameter list containing: **Bytes** **Meaning** **00-07** Node ID of the target session node (if entered with PASSTHRU command) **08-09** Port number in EBCDIC value (if entered with PASSTHRU command) **10-15** Reserved **16-23** The PVM user ID **24-31** Temporary disconnect character string **32-33** Temporary disconnect character string length **34-39** Reserved **40-47** Session termination character string **48-49** Session termination character string length **50-55** Reserved **56-59** Notepad buffer address (if Notepad specified, otherwise empty) **60-63** Reserved **64-67** Console input buffer address **68-71** Reserved **72** Display station model **73-79** Reserved **80-87** Secondary route, if specified **88-90** I/O verification (*ON* or blanks) **91-95** Reserved **96-100** User requested language, if specified **101-103** Reserved
R14	Return address
R15	Entry address.

Register Contents on Return to PVM

Register	Contents
R13	Must be restored to its original value.
R15	One of the following return codes: **0** Normal return **4** Reset Notepad function and initially bypass the Before Read exit. **8** Terminate, go to Before Final exit. **Other** Continue processing, ignore any other return code setting.

Before Read Exit (DVMCUEBR)

This exit, DVMCUEBR, is scheduled prior to checking the console stack to see if there is any data.

Register Contents on Entry to the User Routine

Register	Contents
R14	Return address
R15	Entry address.

Register Contents on Return to PVM

Register	Contents
R13	Must be restored to its original value
R15	One of the following return codes: **0** Normal return **4** Bypass next read. Exit routine is returning an input record. Bypass checking for entry on the console. Routine should set R1 to point to the input record and R0 should contain the input record length. Go to the After Read exit. **8** Terminate, go to the Before Final exit. **Other** Continue processing and ignore any other return code setting.

After Read Exit (DVMCUEAR)

This exit, DVMCUEAR, is scheduled after input has been read from the CMS virtual console or when a return code of 4 is set from the Before Read exit.

Register Contents on Entry to the User Routine

Register	Contents
R1	Address of a parameter list containing: **00-03** Input record length **04-07** Input record address **08-11** One of the following literals is encoded: NORM or SPEC. NORM is used for NORMal readings from the console. SPEC is used for SPECial types of input. Special input types result from reading the input requested by the Before Write Exit.
R14	Return address
R15	Entry address.

Register Contents on Return to PVM

Register	Contents
R13	Must be restored to its original value.
R15	One of the following return codes: **0** Normal return **4** Return to the Before Read exit processing **8** Terminate, go to Before Final exit. **Other** Continue processing and ignore any other return code setting.

Before Write Exit (DVMCUEBW)

This exit, DVMCUEBW, is scheduled prior to writing to the CMS virtual console.

If Notepad is specified with the PASSTHRU command and the After Initialization exit turns off file writes, then the Notepad image goes into a buffer area. If file writes are not turned off, then the Notepad image is also written to the file PASSTHRU DATA.

Register Contents on Entry to the User Routine

Register	Contents
R1	Address of a parameter list containing: **00-03** Output record length **04-07** Output record address.
R14	Return address
R15	Entry address.

Register Contents on Return to PVM

Register	Contents
R13	Must be restored to its original value.
R15	One of the following return codes: **0** Normal return **4** Bypass write operation. The Notepad facility is called to update the Notepad buffer (pointed to by bytes 56-59 of the DVMCUEAI exit parameter list). Update the Notepad file (if active), then schedule the After Write exit. **8** Terminate, go to Before Final exit. **12** Bypass write operation. Perform a read from the virtual console. The read command to use is stored in the first byte position of the output record (the address of which is at bytes 04-07 of the parameter list). The read command is a Read Modified (X'06') or a Read Buffer (X'02'). **Other** Continue processing and ignore any other return code setting.

After Write Exit (DVMCUEAW)

This exit, DVMCUEAW, is scheduled:

- After writing to the CMS virtual console
- After a final response VMCF external interrupt
- For each segment except the last one from PVM
- As a general entry prior to entering into the wait states for external interrupts or console I/O interrupts.

Register Contents on Entry to the User Routine

Register	Contents
R14	Return address
R15	Entry address.

Register Contents on Return to PVM

Register	Contents
R13	Must be restored to its original value.
R15	One of the following return codes: **0** Normal return **4** Perform another write. The exit returns a pointer to a 40 byte VMCF parameter list in R1; which defines a VMCF Send operation from PVM. The DVMUSI module then performs a VMCF Receive using that parameter list. This results in either the transmission of data from PVM or a VMCF return code indicating no further data exists. **8** Terminate, go to Before Final exit. **Other** Continue processing and ignore any other return code setting.

Before Final Exit (DVMCUEBF)

This exit, DVMCUEBF, is scheduled prior to terminating a user session in the CMS virtual machine.

Register Contents on Entry to the User Routine

Register	Contents
R14	Return address
R15	Entry address.

Register Contents on Return to PVM

Register	Contents
R13	Must be restored to its original value
R15	On entry back to CMS, whatever is in R15 is used as a return code.

Rebuilding PVM for Exits

Note: For PVM 2.1.1, see the Program Directory for applying local modifications.

To customize PVM by replacing existing exit routines, the system programmer should follow these steps:

Step 1. Write an exit either as a new module, or as an update at the exit point in the corresponding PVM source module provided as an entry point for the particular type of exit. (See "Exit Point and Module Summary" on page 209 for source module names.) If a new module is used, update the exit point in the source module to branch to the new exit routine. The routine should contain all of the entry points for that type of exit.

See "Exit Point and Module Summary" on page 209 for a table describing module names and entry points.

Step 2. Assemble the module using the VMFASM exec. Access to an appropriate maclib and control file is required. This creates an object

module named *module-name* TEXT, where *module-name* is the filename of the source module. For example, if you enter:

```
vmfasm dvmnue pvm
```

an object module is created for DVMNUE (the session control exit module) using the PVM CNTRL control file. PVM CNTRL lists all the text files used to create the module.

Step 3. Access the minidisk containing the replacement TEXT module above (in the search order) the PVM base code staging disk.

Step 4. **For the Session Control and the Networking Management exits:**

Build the PVM module by issuing the command:

```
pvmbld pvm pvm
```

Link with write access to the PVM system minidisk, and copy the PVM module to it. The PVMBLD command and its options are described in Appendix C, "PVM Commands."

For the CMS User exits:

Build the DVMUSI module by issuing the command:

```
pvmgen dvmusi pvm
```

Link with write access to the system extension minidisk, and copy the DVMUSI module to it. The PVMGEN command and its options are described in Appendix C, "PVM Commands."

PSPI end

Chapter 14. National Languages on Your PVM System

PVM supports several national languages. This chapter describes how to take advantage of these languages.

National Language Overview

The PVM product is shipped with mixed-case American English and uppercase English languages. The system default language is specified on the LANG record in the PVM and PVMG configuration files.

PASSTHRU and MPVM users see PVM messages and menus in the same language being used in their CMS session, provided that PVM is enabled for that language. If a language identifier is specified on the MPVM command, MPVM checks whether it is enabled for the language, and if so, selects it as the default language. If the PVM server is not enabled for this language, a warning message is issued. This means warning messages from the PVM server are issued in its default language. All other messages are in the user's selected language. PVM, MPVM and PVMG users can select the language to be used for a session or for a command response.

You can also order other languages for your PVM system. When you order another language, you receive a feature tape with files for that language on it. These files contain translated information that let users interact with PVM in this new language.

In addition, you can translate PVM messages and menus into local languages or dialects in order to meet the needs of the users at your installation. See "Creating and Using a Message Repository" on page 238 for information on how you can do this.

Languages Available Through IBM

All languages are identified by a 1-to-5 character language ID. The language ID for uppercase English, the default language, is *UCENG*. The IBM-provided languages are:

AMENG American English

GER German

KANJI Japanese

UCENG Uppercase English

Note: Some languages have character sets that require special hardware. Be sure that all display devices in your configuration can properly display the character set of any new language you add to your system. If they cannot, you may be able to create a new language repository that uses the necessary code page.

© Copyright IBM Corp. 1991,1993

Backup National Language

In some instances PVM (and PVMG) cannot use the specified national language when sending or displaying messages and menus. In these cases, PVM and PVMG use a *backup* language.

PVM and PVMG use the backup language:

- As the system national language when an installation does not use the LANG record in the configuration file
- When communicating with PVM systems that are at an earlier release level with respect to the messages being sent
- When sending messages which have been added to PVM between releases to other systems (for example, messages added to PVM by a service update),
- Sending messages to local display stations which do not support the double-byte character set of the national language that would otherwise be used by default.

PVM uses uppercase American English (UCENG) as the backup language. See "Changing the Backup Language" on page 235 for instructions on how to change the PVM backup language.

Installing National Languages for PVM 2.1

Note: All languages are built into the PVM, EFGVIEW, and DVMUSI modules, and the PVMG LOADLIB supplied on the PVM 2.1.1 product tape and built by VMSES/E. For information on installing PVM 2.1.1 NLS features, see the Program Directory.

Installing national languages for PVM involves loading files from the PVM national language feature tape onto system minidisks. The PVM base system must be installed before you can install national languages. The steps you need to take to install national languages for PVM are:

1. Get disks for national language files
2. Load the files from the national language feature tape to disk
3. Enable PVM, MPVM, PASSTHRU (DVMUSI), and PVMG for the additional languages
4. Verify the installation.

Step 1. Getting Disks for National Language Files

You use the MAINT user ID for installing the PVM national languages feature. All of the feature tape files are not loaded onto a single minidisk. Instead, groups of files are loaded onto several of MAINT's minidisks. Lists of the tape files that you will be loading onto the disks can be found in Appendix B, "Tape Layouts for PVM 2.1."

The MAINT minidisks that you will be using are:

Minidisk	MAINT's Address	Comments
Source staging	39E	Optional disk. The source staging minidisk is used for the optional message repository source files. This should be MAINT's 39E disk. It should contain the contents of feature tape file number 5.
Base TEXT staging	49E	Optional disk. The base TEXT staging minidisk is used for the optional message repository object files. This should be the MAINT user ID's 49E disk. It should contain the contents of feature tape file number 6.
PVM system	36E (PVM's 191)	Mandatory disk. The PVM system minidisk is the A-disk for PVM, address 191. We suggest that this also be MAINT's 36E. (The PVM user ID has a multiread link to MAINT's 36E.) It should contain the contents of feature tape file number 3.
System extension	19E	Mandatory disk. This is a minidisk to which users have common read access (such as, the Y-disk extension to the CMS system S-disk). This should be MAINT's 19E. It should contain the contents of feature tape file number 4.
PVMG System	38E (PVMG 191)	Optional disk.
NLS Help Files	401, 402, 405	Optional disks.
MPVM System	38F (PVM 199)	Optional disk.

The minidisk addresses that are referred to in this book are just suggestions as to the addresses that you should use. You are free to choose the addresses that you want. If you choose to use other addresses, remember to consistently use your addresses in place of the suggested ones. However, you should use the suggested addresses if you will be installing the feature tape by the quick method described on page 230. Examples in this book use the suggested addresses.

HELP files have the same file IDs for all languages; therefore, the HELP files for each language installed on your system must reside on a different minidisk on the MAINT user ID. The loading of the HELP files is optional.

The recommended MAINT minidisks for the supported national languages are:

Table 23. Recommended Minidisks for National Language HELP Files

Language	MAINT Minidisk	Feature Tape File Number
Japanese	401	9
Uppercase English	402	10
German	405	11

Step 2. Loading Files From Tape to Disk

The national language feature tape installed by using either the quick method or the standalone method. (These methods are introduced in Chapter 4, "Installing PVM 2.1.")

Note: The standalone method assumes that new minidisks have already been formatted by the CMS FORMAT command. **DO NOT** format any system minidisks that are already in use by the system (for example, MAINT's 19D, 19E, and 401-406).

The two methods are described below.

Loading Feature Tape Files - The Quick Method (Using INSTFPP)

1. The PVM user ID should be logged off so MAINT can get R/W access to its 36E minidisk (this is the same minidisk that PVM links R/W as its 191 A-disk).
2. Log on the MAINT user ID, define storage, and IPL CMS. If you are not already logged on as MAINT, issue the following commands:

   ```
   log on maint
   (Enter password)
   def stor 16m
   ipl cms
   ```

3. Mount the PVM national language feature tape and attach the tape drive to MAINT as 181. Issue the following commands:

   ```
   CP ATTACH raddr MAINT 181
   CP REWIND 181
   ```

   Substitute the tape drive address for *raddr*.

4. Invoke the INSTFPP EXEC with the Product Identification Number. Issue the following command:

   ```
   instfpp 5684NL
   ```

   INSTFPP asks you if you want to load these optional minidisks:

   - PVM SOURCE (MAINT 39E)
   - PVM TEXT (MAINT 49E)
   - PVM/NLS HELP FILES (MAINT 401, 402, 405)
   - PVMG SYSTEM (MAINT 38E/PVMG 191)
   - MPVM SYSTEM (MAINT's 38F - PVM links as 199).

   INSTFPP assumes the following mandatory minidisks will be loaded to the following locations:

   - PVM SYSTEM CODE (MAINT's 36E - PVM links as 191)
   - SYSTEM EXTENSION (MAINT 19E).

   When it asks you if you want to install the PVM National Language Feature tape, respond yes.

   INSTFPP then invokes the PVM installation exec. Answer any questions the installation exec displays. The installation exec completes the installation of the PVM national language feature tape.

Loading Feature Tape Files - The Standalone Method

1. The PVM user ID should be logged off so MAINT can get R/W access to its 36E minidisk (this is the same minidisk that PVM links R/W as its 191 A-disk).
2. Log on the MAINT user ID, define storage, and IPL CMS. If you are not already logged on as MAINT, issue the following commands:

   ```
   log on maint
   (Enter password)
   def stor 16m
   ipl cms
   ```

3. Mount the PVM distribution tape and attach the tape drive to MAINT as 181. Issue the following commands:

   ```
   CP ATTACH raddr MAINT 181
   CP REWIND 181
   ```

   Substitute the tape drive address for *raddr*.

4. Issue the following two commands:

   ```
   vmfplc2 scan i5684nls memonl (eot
   vmfplc2 fsf 1
   ```

   to move tape to correct position for the source files for the PVM national language feature.

5. Load the PVM SYSTEM files. This step is mandatory. Do not bypass this load. Enter:

   ```
   access 36e a
   vmfplc2 load * * a
   ```

   36E is the address of your PVM system.

6. Load the SYSTEM EXTENSION files. This step is mandatory. Do not bypass this load. Enter:

   ```
   access 19e a
   vmfplc2 load * * a
   ```

   19E is the address of your SYSTEM EXTENSIONS.

7. If you do not want to load the PVM source files, then enter:

   ```
   vmfplc2 fsf 1
   ```

   and go on to the next step.

   To load the source files, enter:

   ```
   access 39e a
   vmfplc2 load * * a
   ```

   39E is the address of your Message Repository Source files.

8. If you do not want to load the PVM TEXT files, then enter:

   ```
   vmfplc2 fsf 1
   ```

   and go on to the next step.

   To load the TEXT files, enter:

   ```
   access 49E a
   vmfplc2 load * * a
   ```

   49E is the address of your Message Repository Object files.

9. If you do not want to load the PVMG System files, enter:

   ```
   vmfplc2 fsf 1
   ```

   and go to the next step.

   To load the PVMG System files, enter:

   ```
   access 38E a
   vmfplc2 load * * a
   ```

   38E is the address of your PVMG System files.

10. If you do not want to load the MPVM System files, enter:

    ```
    vmfplc2 fsf 1
    ```

    and go on to the next step.

    To load the MPVM System files, enter:

    ```
    access 38F a
    vmfplc2 load * * a
    ```

    38F is the address of your MPVM System files.

11. If you do not want to load the Japanese HELP files, enter:

    ```
    vmfplc2 fsf 1
    ```

    and go on to the next step.

    To load the Japanese HELP files, enter:

    ```
    access 401 a
    vmfplc2 load * * a
    ```

12. If you do not want to load the uppercase English HELP files, enter:

    ```
    vmfplc2 fsf 1
    ```

    and go on to the next step.

    To load the uppercase English HELP files, enter:

    ```
    access 402 a
    vmfplc2 load * * a
    ```

13. If you do not want to load the German HELP files, enter:

    ```
    vmfplc2 fsf 1
    ```

    and go on to the next step.

    To load the German HELP files, enter:

    ```
    access 405 a
    vmfplc2 load * * a
    ```

14. Enter:

    ```
    cp rewind 181
    cp detach 181
    ```

    to rewind the tape and detach it from MAINT.

Step 3. Enabling PVM Components for Additional Languages

If you installed all of the languages from the feature tape, go to "Step 4. Verifying National Language Installation."

Otherwise:

For PVM:

1. For each language you installed, edit the PVM EXEC file and ensure that it lists only the languages that you want enabled. You may have to add or delete DVM*langid* statements.
2. Rebuild PVM using the PVMBLD command. See "PVMBLD" on page 315 for information on how to use PVMBLD.

For MPVM:

1. For each language that you installed, edit the EFGVIEW EXEC file and ensure that it lists only the languages that you want enabled. You may have to add or delete EFG*langid* statements.
2. Rebuild EFGVIEW using the MPVMBLD command.

For PVMG:

1. For each language that you installed, edit the PVMG LKEDCTRL file and ensure that it lists only the languages that you want enabled. You may have to add or delete INCLUDE DVM*langid* statements.
2. Rebuild PVMG using the PVMGBLD command.

For DVMUSI (PASSTHRU):

1. For each language you installed, edit the DVMUSIE EXEC file and add or delete *langid* statements in the form EFG*langid*.
2. Rebuild DVMUSI using the PVMGEN command. See "PVMGEN (PVM 2.1)" on page 317 for information on how to use PVMGEN.

Place LANG and XDBC records in the PVM and PVMG configuration files, if applicable. See Appendix D, "PVM Configuration File Records" for information on the LANG and XDBC records for PVM, see Appendix F, "PVMG Configuration File Records" on page 401 for these records for PVMG.

Step 4. Verifying National Language Installation

For PVM

To verify the installation of the PVM national language feature tape :

1. Log on to the MAINT user ID.
2. Enter:

   ```
   cp detach 36e
   ```

   This ensures that PVM can link to 36E as its 191 in R/W mode.
3. Log on to your PVM user ID.

4. After a short time, you should see this at your display station:

   ```
   Execution begins...
   VM Pass-Through Facility 2.1.0000, built MM/DD/YY HH.MM.SS, is ready
   :
   ```

5. Enter:

   ```
   query lang
   ```

   If you installed all of the languages from the feature tape, PVM should respond that PVM is enabled for all four supported national languages (mixed-case American English, Japanese, uppercase English, and German).

   If you chose to install only selected languages, PVM should respond that it is enabled for the languages you specified in "Step 3. Enabling PVM Components for Additional Languages."

6. To disconnect from the PVM user ID, enter:

   ```
   disc
   ```

7. From a locally attached display station, enter:

   ```
   dial pvm
   ```

8. A PVM Selection Menu should be displayed. Change the LANGUAGE field to an enabled language ID.

9. Type in a node ID that is not valid in the DESTINATION field and then press ENTER. You should get an error message in the language you specified.

Verification is now complete. Press PA1 to exit from the PVM Selection Menu.

For MPVM

This verification example assumes that PVM is already running, and that you are logged on to a virtual machine.

1. Enter

   ```
   mpvm kanji
   ```

2. The MPVM Selection Menu should be displayed containing the nicknames that were defined in the MPVM NAMES file, located on this user ID's A-disk. Names found in the MPVMSYS names file located on the MPVM common disk will appear on this menu.

3. Enter an incorrect nickname. You should get an error message in Japanese.

Verification is now complete. Press PF3 or PF12 to exit from the MPVM Selection Menu.

For PVMG

To verify the national language installation for PVMG:

1. Enter:

   ```
   cp detach 38E
   ```

   from the MAINT user ID. This will ensure that PVMG can link to this disk as its 191 in R/W mode.

2. Log on to user ID PVMG

3. This will invoke IPL GCS, which invokes the PROFILE GCS. You should see a dsiplay similar to this one:

   ```
   VM Pass-Through Gateway Services 2.1.0000, built
   MM/DD/YY HH.MM.SS, is ready
   ```

4. Enter:

   ```
   pvmg query lang
   ```

   If you installed all of the languages from the feature tape, PVMG should respond that PVMG is enabled for all four supported national languages (mixed-case American English, Japanese, uppercase English, and German).

   If you chose to install only selected languages, PVMG should respond that it is enabled for the languages you specified in "Step 3. Enabling PVM Components for Additional Languages."

5. Enter

   ```
   pvmg disc
   ```

   to disconnect the PVMG user ID.

6. Enter

   logon *applid*(*xxxxxxxx*)

   (*xxxxxxxx* is PVMG's applid)

7. A PVMG Selection Menu should be displayed. Change the LANGUAGE field to an enabled language ID.

8. Type in a node ID that is not valid in the DESTINATION field and press ENTER. You should get an error message in the language you specified.

Verification is now complete. Press PA1 to exit from the PVMG Selection Menu.

Changing the Backup Language

When changing the PVM, MPVM, PASSTHRU (DVMUSI), or PVMG backup language, first consider these points:

- The backup language should be the language that is most universally understood by all of the users throughout the PVM network.
- The backup language you choose must be an enabled PVM or PVMG language. It should also be the same language at every system in the PVM network.
- Do not use a language that contains double-byte characters as the backup language unless all display stations in your PVM network support those characters.

Follow the steps described below to change the backup language. The steps are performed while you are logged on to the MAINT user ID.

1. Enter these two commands:

   ```
   pvmcmp dvmnlangid net
   pvmcmp efgnlangid net
   ```

   where *nlangid* is the language ID of the new backup language.

   Note: Use the CTL option of the PVMCMP command if there are any update files for PVM message repositories or for the PVMDEST FILE.

2. Enter these two commands:

   ```
   pvmcmp dvmolangid nat
   pvmcmp efgolangid nat
   ```

   where *olangid* is the language ID of the old backup language.

   Note: Use the CTL option of the PVMCMP command if there are any update files for PVM message repositories or for the PVMDEST FILE.

   Note: See the Program Directory regarding building PVM 2.1.1 and applying local modifications.

3. Rebuild PVM using the PVMBLD command. See "PVMBLD" on page 315 for information on how to use PVMBLD.
4. Rebuild DVMUSI using the PVMGEN command. See "PVMGEN (PVM 2.1)" on page 317 for information on how to use PVMGEN.
5. Rebuild PVMG using the PVMGBLD command. See "PVMGBLD (PVM 2.1)" on page 387 for more information on how to use the PVMGBLD command.
6. Rebuild MPVM using the MPVMBLD command. See "MPVMBLD (PVM 2.1)" on page 307 for information on how to use MPVMBLD.

Changing Message Destinations

PVM and PVMG use a file named PVMDEST FILE to determine the destinations for PVM messages. (The PVMDEST FILE is on the PVM source staging minidisk.) Records in this file map a message number to keywords describing the message destinations. Keywords in the file include:

OP Indicates that the message is sent to the CP operator (the OPID configuration file record)

CONS Indicates that the message is sent to the PVM console (PVM user ID) or to the PVMG console (PVMG user ID)

For PVM, CONS also indicates that the message is sent to the audit file.

ORIG Indicates that the message is sent to the command originator for command responses.

PROP Indicates that the message is sent to the PVMPROP virtual machine (PVMPROP configuration file record)

For example, the PVMDEST FILE record:

```
0209      OP CONS      PROP
```

Indicates that message 209 is sent to the CP operator, the PVM user ID, and to the PVMPROP user ID (if one exists).

The PVMDEST FILE record:

```
1209      OP CONS      PROP
```

Indicates that message 1209 is sent to the CP operator, the PVMG user ID, (messages numbered 1000 - 1999 are PVMG messages) and to the PVMPROP user ID (if one exits).

To change the destinations for a PVM or PVMG message, follow these steps:

1. Change message entry in the PVMDEST FILE.
2. Compile all of the message repositories used by the PVM MODULE (that is, all of the DVM*langid* REPOS files). Use the PVMCMP command described on page 316. Use the *NAT* option for all repositories, except for the backup language repository; use the *NET* option when compiling the repository for the backup language. (You can create your own update file using CMS UPDATE. Use the CTL option of the PVMCMP command.)

   Note: See the Program Directory regarding building PVM 2.1.1 and applying local modifications.
3. Rebuild PVM using the PVMBLD command. See page 315 for a description of the PVMBLD command.
4. Rebuild PVMG using the PVMGBLD command. See "PVMGBLD (PVM 2.1)" on page 387 for a description of the PVMGBLD command.

Changing the PVM Translate Table

PVM uses a 256-byte translate table to translate lowercase American English alphabetic characters into uppercase when reading input from a PVM menu. If you need to modify the translation algorithm for the needs of another language, or to accommodate the use of a different code page, follow these steps:

1. Change the translate table located at the label DVMTRN in the DVMTBL ASSEMBLE file. (The DVMTBL ASSEMBLE file can be found on the PVM source staging minidisk.)
2. Recompile DVMTBL ASSEMBLE file using VMFASM. Enter:

   ```
   vmfasm dvmtbl pvm
   ```

   where `pvm` indicates the PVM CNTRL file that is distributed with PVM.

   Note: See the Program Directory regarding building PVM 2.1.1 and applying local modifications.
3. Rebuild PVM using the PVMBLD command. See "PVMBLD" on page 315 for information on how to use PVMBLD.
4. Rebuild DVMUSI using the PVMGEN command. See "PVMGEN (PVM 2.1)" on page 317 for information on how to use PVMGEN.
5. Rebuild PVMG using the PVMGBLD command. See "PVMGBLD (PVM 2.1)" on page 387 for information on the PVMGBLD command.
6. Rebuild MPVM using the MPVMBLD command. See "MPVMBLD (PVM 2.1)" on page 307 for information on the MPVMBLD command.

Creating and Using a Message Repository

At some time, you may want to change the PVM, PVMG, PASSTHRU (DVMUSI) and MPVM message repositories, or create your own. You can change the message repository to:

- Change the wording of messages or menus
- Translate messages to another language or dialect
- Change the format of time stamps on messages.
- Change only certain characters so that they display properly with the code page used in your country.

  For example, in the United Kingdom, you might wish to copy the nnnAMENG repository and alter the copy so that it will display characters properly using the United Kingdom's code page.

The steps involved with creating and then using your own message repository or changing a PVM message repository are:

1. Print and read *Section One* of the READTHIS FIRST file found on the CMS system extension minidisk (default 19E)
2. Create a message repository file using XEDIT
3. Compile the repository
4. Verify the accuracy of the message file
5. Make the repository file available.

Creating a Message File

There are two sets of message repositories in PVM — one for the PVM MODULE and the PVMG MODULE, and one for the DVMUSI and EFGVIEW MODULEs that run in a PVM user's CMS session when the PASSTHRU or MPVM commands are used. Messages with identifiers for PVM and PASSTHRU (DVMUSI) range from 0-999. Messages with identifiers for MPVM and PVMG range from 1000 - 1999. Repositories containing definitions for messages and screens used by the PVM MODULE and PVMG MODULE are named DVM*langid* REPOS, where *langid* is a 1-to-5-character language identifier. Repositories containing definitions for messages used by the DVMUSI MODULE and EFGVIEW MODULE are named EFG*langid* REPOS.

When you create a message repository file, you must follow certain notation rules. These rules are explained in "Rules for Making Your Own Repository" on page 240.

You can use one of the IBM-supplied message files as a reference when you create your own message file. (Use the CMS UPDATE command to make sure that you are using the most current level of the IBM-supplied message file.)

Compiling the Message File

Note that some message text may be identical with two different messages. For example, messages 0071 and 1071 contain identical message texts. You should ensure that your new message text for messages that contain identical message text (like 0071 and 1071) is identical with your new message repository.

You can compile the message file by using the PVMCMP command. Enter:

```
pvmcmp repository
```

where *repository* is the filename of the message repository you created in XEDIT.

Note: Use the CTL option of the PVMCMP command if there are any update files for the file PVMDEST FILE.

PVMCMP displays error messages at your display station and in the listing file for any invalid statement it finds in the message file. If you have any errors, just XEDIT the file and correct any mistakes. The listing file has the same filename as the repository file and a filetype of *$LISTING*.

When your message repository does not contain syntax errors, PVMCMP creates an output file with the same filename as the input file and a filetype of *TEXT*. This TEXT file contains the machine readable form of the message file that the PVM message processor recognizes.

See page 316 for details on the PVMCMP command.

Verifying the Message File

To verify that your message and screen definitions are functionally accurate:

1. The DVM*langid* $COMP*nnn* file should be identical with the IBM-supplied DVMUCENG $COMP*nnn* file.
2. The EFG*langid* $COMP*nnn* file should be identical with the IBM-supplied EFGUCENG $COMP*nnn* file.
3. The DVM*langid* $SYNC*nnn* and the EFG*langid* $SYNC*nnn* files should be identical.

You can use the CMS COMPARE command to verify that these files are identical. See "PVMCMP" on page 316 for more information about the files created when a message file is compiled.

Making the Message File Available

Once the message file has been compiled, you must make the message file available for use. To do this, perform these steps:

1. For each language file you created, update the DVMLNG ASSEMBLE file found on the PVM source staging minidisk. Add the language ID to the line in the file that looks like this:

   ```
   LANGS IDS=(AMENG,UCENG,GER,KANJI)
   ```

   For example, to add the language identified by XYZ, you could have:

   ```
   LANGS IDS=(AMENG,UCENG,GER,KANJI,XYZ)
   ```

2. Recompile DVMLNG ASSEMBLE file using VMFASM. Enter:

   ```
   vmfasm dvmlng pvm
   ```

   where `pvm` indicates the PVM CNTRL file.

3. For each language you installed, add a line in the PVM EXEC file for PVM 2.1 or in the PVMMOD EXEC file for PVM 2.1.1 in the form DVM*langid*.

   Note: See the Program Directory regarding building PVM 2.1.1 and applying local modifications.

4. Rebuild PVM using the PVMBLD command. See "PVMBLD" on page 315 for information on how to use PVMBLD.
5. For each language you installed, add a line in the DVMUSI EXEC file in the form EFG*langid*.
6. Rebuild DVMUSI using the PVMGEN command. See "PVMGEN (PVM 2.1)" on page 317 for information on how to use PVMGEN.

   Note: You need not perform the next two steps if you have not installed PVMG.
7. For each language you installed, add the following line in the PVMG LKEDCTRL file:

   INCLUDE DVM*langid*.
8. Rebuild PVMG using the PVMGBLD command. See "PVMGBLD (PVM 2.1)" on page 387 for information regarding the PVMGBLD command.
9. For each language you installed, add a line to EFGVIEW EXEC, in the form, :EFG*langid*.
10. Rebuild MPVM using the MPVMBLD command. See "MPVMBLD (PVM 2.1)" on page 307 for information on how to use the MPVMBLD command.

Rules for Making Your Own Repository

There are five types of records in a repository file:

- Comments
- Master
- Time stamps
- Message and screen definitions
- Control lines.

Blank lines are also allowed in a repository file.

Control lines should not be changed when creating your own repository. The control lines are used to increment the compatibility level for subsequent releases and updates. Compatibility levels are used by PVM and PVMG for message level synchronization in the network. For example, if a PVM Version 2 node needs to send a PVM Version 2 message to a PVM 1.4 node, the message is sent to the PVM 1.4 node in the network (backup) language. If the message is a PVM 1.4 message, the message is sent in encoded format.

Comment Record

A comment record begins with an asterisk in column 1. Comment records can be placed anywhere in the message repository and should describe what is in the file. You can use columns 1 through 63 for your comments.

When you create your file using XEDIT, you must make the comment record in the following format:

```
=====       .
=====       .
===== * This is a sample comment record
      |...+....1....+....2....+....3....+....4....+....5....+....6...
=====       .
=====       .
```

Master Record

The master record defines the characters that have special meanings in the repository's message and screen records. This record is mandatory. You can use columns 1 through 63 for the master record. Also, it must be the first line in the repository other than blank lines or comments. All characters used on the master record must be a single-byte character whose hexadecimal value is between X'40' and X'FE', inclusive. When you create your file using XEDIT, you must make the master record in the following format:

```
===== * * * Top of File * * *
===== a bcd y z
      |...+....1....+....2....+....3....+....4....+....5....+....6...
=====       .
=====       .
=====       .
```

a Defines a character that identifies a substitution variable in a message record, or indicates the beginning of a highlighted field in a screen record. The character you use cannot be a digit and must be followed by at least one blank character.

This character is called the *start variable* character for message definitions and the *start highlighting* character for screen definitions.

bcd
Defines the characters indicating the severity of numbered messages. For example, in the IBM-supplied mixed-case and uppercase American English message files, these are:

Code	*Message Type*
E	Error
I	Information
W	Warning
S	Severe Error (MPVM only).

At least one severity character must be specified. Duplicate characters and blank characters between message types are not allowed. The last message type specified must be followed by at least one blank character.

y Specifies the character that is used at the end of a line when it is necessary to continue the definition of a message or screen record on an additional line in the repository. This character must be followed by at least one blank. The character selected must not be a digit.

This character is called the *continuation* character.

z Indicates the position of a required blank in a message record or the end of a highlighted field in a screen record. The character specified must not be a digit. This character is optional.

This character is called the *end highlighting* character in screens, and the arbitrary blank character in messages.

Here is a sample master record:

```
$ EIW # ¢
```

Time Stamp Records

There are two time stamp records. Time stamp records are used only in the DVM*langid* message files. When you create your file using XEDIT, make your time stamp records in the following format:

```
=====        .
=====        .
===== AMSTAMP ---------------- format ----------------------------
===== PMSTAMP NN ------------- format ----------------------------
      |...+....1....+....2....+....3....+....4....+....5....+....6....+....7...
=====        .
=====        .
```

AMSTAMP
Defines the time stamp format for PVM time stamps issued before noon. The character string *AMSTAMP* must be in columns 1-7.

PMSTAMP
Defines the time stamp format for PVM time stamps issued after noon. The character string *PMSTAMP* must be in columns 1-7.

NN
Indicates what type of clock should be used for time stamps. One of two clock types must be specified on the PMSTAMP record. **NN** can be:

12 Telling PVM to use a 12-hour clock.

24 Telling PVM to use a 24-hour clock.

format
Specifies the format of the time stamp. This field starts in column 9 for AMSTAMP and in the first nonblank character column following the **NN** field for PMSTAMP. A format is required on the AMSTAMP record and can be omitted on the PMSTAMP record. If a format is not specified on a PMSTAMP record, then the format specified on the AMSTAMP record is used. Leading, trailing, and excess intervening blanks are removed from the format, and the display length of the format, after removal of these blanks, must not exceed 40 characters. The format must not proceed beyond column 63.

The following keywords can be used to specify a time stamp format. The keywords must appear in uppercase when used in the time stamp record. At least one keyword must be specified.

Keyword	Use
MO	The month (2 digits)
DD	The day (2 digits)
YY or YYYY	The year. If YYYY is specified, the 2-digit year will be preceded by *19* (that is, the equivalent of coding *19YY*).
HH	The hour (2 digits)
MI	The minute (2 digits)
SS	The second (2 digits)

Note: Each keyword can be used only once in the time stamp record. Except for keywords, all characters used in the time stamp format appear in PVM time stamps. Do not use character strings in time stamp records that exactly match a keyword. For example, if you specified this PMSTAMP:

```
PMSTAMP 12 HOUR: HH  MINUTE: MI  SECOND: SS
```

PVM tells you that this record has an error in it because the *MI* keyword is used twice; once in *MINUTE:* and then again after *MINUTE:*. The following PMSTAMP record would not contain any errors:

```
PMSTAMP 12 Hour: HH  Minute: MI  Second: SS
```

because the *MI* keyword appears only once.

Message and Screen Records

Message and screen records define three types of information:

Numbered messages messages that are displayed with headers that contain a three digit message identifier.

Unnumbered messages messages that are displayed without headers, usually in the message display areas of PVM display menus but occasionally embedded in numbered messages.

Screen items items that collectively define PVM display menus.

Each of these records in a repository file contains five fields. When you create your file using XEDIT, you must make every message and screen record in the following format:

```
=====       .
=====       .
===== NNNNVVLLS ------------------------ text --------------------
      |...+....1....+....2....+....3....+....4....+....5....+....6....+....7...
=====       .
=====       .
```

NNNN
is the message or screen identification number, in columns 1-4. Repository numbers:

0000-1999 are used for numbered messages. You must use a 4 digit message number in a message repository, but only the last three digits are displayed in the message header. (Message numbers 000 - 999 are for PVM and (DVMUSI), and messages 1000 - 1999 are for PVMG and MPVM. PVM and DVMUSI use the prefix DVM. PVMG and MPVM use the EFG prefix.)

2000-6999 are not used.

7000-7999 are used for unnumbered messages whose output length is limited to 80 bytes. These include the messages that must fit into the message display areas of PVM display menus. Each message in the 7000 to 7999 range must have a counterpart message in the 8000-8999 range.

8000-8999 are used for unnumbered messages whose output length is limited to 48 bytes. These are messages that must fit into the display screens of PVM Release 3 and earlier. Each message in the 8000 to 8999 range must have a counterpart message in the 7000-7999 range. PVM always uses the 8000-8999 message when it must send a message to any remote node that is at an earlier release level with respect to that message.

9000-9999 used for screen records. The 4 digit numbers for screen definitions are not displayed.

VV Is the message format, in columns 5-6. This field is for a message that can be in several versions. If a message has just one format, you do not need to type anything, and the format field defaults to *01*. You cannot use *00* as a format number. This must be blank for screen records.

LL is the line number of the message, in columns 7-8. This field is used if you want the text for a single message format to be displayed on more than one line. Messages that spread across more than one line must have sequential, consecutive line numbers.

If a single format of a message has only one line, you do not need to type a line number. The line number defaults to *01*. You cannot use *00* as a line number.

Both digits must be specified. The range of valid line numbers is *01* to *15*.

The line number must be blank for records defining either PVM unnumbered messages or screen records.

s is the severity code, in column 9. The severity codes defined in the IBM-supplied mixed-case and uppercase American English message files are:

Code	*Message Type*
E	Error
I	Information
W	Warning
S	Severe Error (MPVM only).

text

is the message text, starting in column 11. PVM removes any leading, trailing, and excess intervening blanks in the text field. You can specify characters up to column 63; use the continuation character defined in the master record to continue on the next line of the message file. To make a continuation line, use the continuation character as the last nonblank character in the text field. The continuation line must be the next record in the repository that is neither a comment nor a blank record. Columns 1-10 must be blank on the continuation record.

The continuation character must not be used as the first nonblank character in the text.

Also, you can use substitution variables in your text definitions; see "Substitution Variables in Messages" on page 245 for more information.

The display lengths of messages, after substitution for variables, is limited as follows:

Messages 0001-1999 Must not exceed 140 characters including the message header. There are some exceptions to this rule listed in "Special Considerations for Message and Screen Records" on page 246.

Messages 7000-7999 Must not exceed 80 characters.

Messages 8000-8999 Must not exceed 48 characters.

The following table Table 24 summarizes the contents of message and screen definition records.

Table 24. Message and Screen Definition Record Summary

Columns	Numbered Messages	Unnumbered Messages	Screen Items
1-4	Identification number (0000-1999)	Identification number (7000-8999)	Identification number (9000-9999)
5-6	Version number	Version number	Blank
7-8	Line number	Blank	Blank
9	Severity code character	Blank	Blank
10	Blank	Blank	Blank
11-63	Message Text	Message Text	Screen Text
Note: Columns 1-10 must be blank in continuation lines.			

Substitution Variables in Messages

PVM uses substitution variables in message repository files for information it does not know until execution time. Substitution variables are replaced with actual values by the PVM message handler at execution time. Modules that call the message handler to construct a message pass an argument list that contains the values for the substitution variables. You cannot use substitution variables in screen definition records.

A substitution variable is a five-character string in the following format:

AMMNN

A Is the start variable character defined in the repository's master record.

MM
Identifies the parameter that is to replace this particular substitution variable. For example, *02* identifies the second parameter in the argument list. The valid range of values for MM is *01* to *51*. Both digits must be specified. Parameter positions cannot be skipped (for example, if you have position *03*, then there must also be positions *01* and *02*).

NN
Indicates the maximum length in characters of the replacement string. The valid range of values for NN is *01* to *99*.

For example, if *$* is defined as the start variable character, then *$0205* defines a variable that is to be replaced by the second parameter in the argument list to a maximum length of five characters.

If you change the text of a message, or translate a message into another language, all of the substitution variables used in the original message must be present in the new message. The substitution variables themselves cannot be changed. They can, however, be placed in any order within the text.

Special Considerations for Message and Screen Records

- You can use highlighting to enhance the appearance and readability of PVM display screens. To use highlighting you must define an end highlighting character in the master record.

  To define a highlighted field, use the start highlighting character followed by the field of characters that is to be highlighted, followed by the end highlighting character. The start highlighting character does not count toward the overall length limit of the screen item if it is the first character in a text definition. The end highlighting character does not count toward the overall length limit of the screen item if it is the last character in a text definition.

  Highlighting works only for screen records 9001, 9074, and 9075. Highlighting characters in any other screen record are converted to blanks.

  For example, suppose you defined *$* and *¢* as the start highlighting and end highlighting characters. If you specified *$ENTER¢* in your text definition, then the word *ENTER* is highlighted in the screen display.

  Certain lines on the PVM Selection and Initial Menus are highlighted without regard to highlighting characters in the screen item definitions. When this occurs, any highlighting characters are converted to blanks.

  You cannot use highlighting in message records.

- Some PVM messages have special restrictions you must follow. These messages and their restrictions are listed in the following table.

Message	Restrictions
0000	• Must be defined with only one version and as single-line messages. • The text definition must contain exactly one substitution variable. • The length of the text of this message after variable substitution cannot exceed 79 characters.
0010, 1010, 0050, 1050, 0068, 1068	• Must be defined with only one version and as single-line messages. • Do not specify any text definition on the message record for these messages.
0127, 1127	• Must be defined with only one version and as single-line messages. • The text definition for this message must contain one substitution variable. Additional variables and characters are not allowed.

- When creating your own message repository, there are limits to the display length for screen records. The display length limits for screen records are:

	Display Length Limit				
	4	**5**	**27**	**67**	**79**
Screen Record Identifier	9010-9017	9018-9020	9002-9007, 9073, 9076	9008	9000-9001, 9009, 9074-9075, 9100-9114

- Screen records 9101-9114 define the text of the PVM I/O error recovery menu. Each screen record defines a single line of text. The text consists of a single paragraph beginning with record 9101. The records for unneeded display lines must be defined with blank text fields. The first record with a blank text field indicates the end of the paragraph.
- Blanks between words in screen record 9009 are not removed by PVM and are left as defined. Leading and trailing blanks are removed.
- Screen record 9021 defines the five characters that are in the PVM Selection Menu to indicate node types. The text field of this record must contain exactly five nonblank, single-byte characters. Duplicate characters are not allowed.
- Screen record 9022 defines a padding string that is used in the PVM Selection Menu to enhance the readability of certain lines. The text field is limited to a maximum of two, nonblank characters. PVM English language repositories have −> defined as the padding string. The padding string is used in screen records 9002-9008.

  If only one character is specified as a padding string, it is treated as though the same character is specified twice. For example, = is treated as if == had been specified.
- Records in the range 7510-7519 are reserved for use in the MPVM Selection Menu (which is built at MPVM initialization). Records 8510-8519 are reserved as counterpart messages.

  The PVM compiler doesn't check to see that there are no variables in these records. It is the programmer's responsibility to ensure that no variables in these messages are used for screen lines by MPVM.

Control Lines

Control line records are used in message repositories to separate messages for different versions and releases of PVM. Messages must remain in the correct control section of the file. The control line sections are used for message level synchronization in the PVM network. For example, any messages in the section below

```
CONTROL CL2
```

are PVM CSE Update messages.

Any messages below

```
CONTROL CL3
```

are for PVM Version 2.

Messages that may be added for service are added after the

CONTROL SERVICE

record.

Message Repository Samples

Figure 37 illustrates the records defining three sample messages.

```
|...+....1....+....2....+....3....+....4....+....5....+....6...
$ EIW # ¢
.
.
.
09993201I Port $0102 is a $0204, Model $0302, currently in #
          session.
09993202I It is a color terminal with extended highlighting
09993203I and programmable    symbols.  The last status rece#
          ived
09993204I from this terminal was $0404.

7006      $0108 $0208 link is already in use.
8006      $0108 $0208 LINK IS ALREADY IN USE.
```

Figure 37. Example Message Records

When it is displayed, sample message 999I could look like this:

```
DVMQRY999I Port 20 is a 3279, Model 3, currently in session.
DVMQRY999I It is a color terminal with extended highlighting
DVMQRY999I and programmable symbols. The last status received
DVMQRY999I from this terminal was 4050.
```

Figure 38, and Figure 39, are samples of the screen definition records for the PVM Selection, Initial, and I/O Error Recovery Menus.

```
|...+....1....+....2....+....3....+....4....+....5....+....6...
9000      VM/PASS-THROUGH FACILITY
9001      You Can Select a Node With the Cursor and Press #
          $ENTER¢
9002      Destination
9003      Language
9004      End Session
9005      Verify
9006      Port
9007      Route
9008      Your Identification
9009      PF8= Scroll              CLEAR key = Top Sc#
          reen                   PA1= Exit
9010      PF1=
9011      PF2=
9012      PF3=
9013      PF4=
9014      PF5=
9015      PF6=
9016      PF7=
9017      PF9=
9018      PF10=
9019      PF11=
9020      PF12=

*   9021 = characters displayed beside certain node names:
*     1st - the local PVM system.
*     2nd - directly linked PVM nodes.
*     3rd - directly linked non-PVM systems.
*     4th - directly linked ROCF nodes.
*     5th - name of a line group.

9021      L N S R G

9022      ->
9073      Local Node
9074      Press $ENTER¢ for a Session With the Following #
          Options
9075      Press $CLEAR¢ to View the Selection Screen
9076      Address
```

Figure 38. Initial and Selection Menu Definitions

```
|...+....1....+....2....+....3....+....4....+....5....+....6...
9100      PROGRAM ERROR RECOVERY
9101      The program you are using through PVM has #
          transmitted
9102      control information that is not compatible with your
9103      display unit.  At this time you can attempt #
          to recover
9104      from the error by pressing keys that are appropriate
9105      to the program you are using, or you can end your
9106      PVM session.  The ENTER key will end your PVM #
          session,
9107      unless you specified a PF or PA key for that purpose.
9108
9109
9110
9111
9112
9113
9114
```

Figure 39. I/O Error Recovery Menu Definitions

Chapter 15. Facilities for Automating PVM and PVMG Operations

GUPI

Application programming facilities available to the system programmer or system operator are:

- MPVM macros and commands (see the *VM/Pass-Through Facility Programming Reference* for this information).
- The command response interface (CRI) discussed in this chapter.
- A programmable operator (PVMPROP) message interface discussed in this chapter.
- Application-to-application communication using IUCV protocols (see Chapter 16, "Application-to-Application Overview" on page 259 for this information).

You can use these facilities to write application programs that increase the efficiency of system and network operations.

The programmable operator facility allows you to develop applications for the remote automated operation of PVM systems. Using the programmable operator message interface, your application can perform tasks that do not require an on-site operator.

A sample exec, PVMSNAP EXEC, is provided to show how an application uses PVMPROP/CRI. See "Sample Application" on page 257 for more information about this example application.

PVMPROP - A Programmable Operator Interface for PVM and PVMG

The PVMPROP configuration file record identifies a user ID that receives copies of error messages issued by PVM or PVMG. A PVMPROP user ID can be used to run an application program that acts like a PVM or PVMG operator. Your application program can process PVM or PVMG error messages and monitor PVM for problem indicators and then attempt to correct a problem or notify the real operator of a problem.

PVMPROP can also use the Command Response Interface (CRI), described in the next section. Command prefixes are discussed later in this chapter.

Error messages sent to a PVMPROP user ID by PVM or PVMG are sent using the CP SMSG command using the PVMPROP/CRI message protocol.

For example, PVM issues message 137 when there are no free buffers available to a PVM task. This situation can sometimes cause severe performance degradation of the PVM system. A PVMPROP user ID receives a copy of message 137. An application program running on that user ID could be programmed to notify the system programmer of the situation.

In another scenario, a PVM link goes down, and PVM issues a message. One copy goes to the AUDIT file, and the other is sent to the PVMPROP user ID. An installation written application intercepts the message and sends a command using the *T.* prefix of the CRI to PVM telling PVM to START the link that is down.

© Copyright IBM Corp. 1991,1993

If successful, the application ends. If the START command is unsuccessful, as determined from the command response, the application could send a message to the system operator, along with specific instructions on what corrective action to take.

You can use the PVMDEST FILE on the PVM system minidisk to find out which messages PVM or PVMG sends to a PVMPROP user ID. Each line in PVMDEST FILE lists a PVM message number and destinations for that message. The character string *PROP* on a line in the file indicates that the message is sent to a PVMPROP user ID. See "Changing Message Destinations" on page 236 for more information on this topic.

The Command Response Interface

Your applications can use the PVM *T.* command prefix to tell PVM or PVMG to send command responses to you in a language- independent format that is easily understood by applications. (For information about using command prefixes, see "Using Command Prefixes" on page 255.) PVM uses the CP SMSG command to send a command response when the *T.* prefix is used.

PVM and PVMG encode the command responses to *T.* prefixes. This encoded message is language-independent and can be easily processed by an application program. See "PVMPROP/CRI Message Protocol" for a description of the information in an encoded message.

PVM sends message number 0001 to indicate the end of the command response. This makes it easier for an application program to determine when the responses for a command are finished.

The *M.* and *S.* prefixes can be used:

- To find out which messages PVM issues in response to a command
- When an application needs to route the natural language message to a human operator.

For example, you can write an application that queries PVM or PVMG periodically for network status. If the application uses the *T.* prefix , PVM or PVMG respond in a *tokenized* form that is easily interpreted by the application. If the application uses the *M.* or *S.* prefixes, PVM or PVMG responds in natural language. These responses can be routed to a human operator.

PVMPROP/CRI Message Protocol

PVM and PVMG use the PVMPROP/CRI message protocol format when:

- The command response interface is used
- PVM or PVMG error messages are to be copied to a PVMPROP user ID.

The message format consists of a 30 byte header, which is followed by an encoded message, as follows:

PVM␣␣*nnnn*␣00*ll*␣*nodename*␣*yyyyyy*␣message

(Note that each ␣ symbol represents a blank space).

Message Format

The PVMPROP/CRI message format consists of a 30-byte header (containing five parts) followed by an encoded message. Each part is separated by a blank, making it easier for application programs to process the response. The table that follows describes the header, and the message text that follows.

Table 25. Parts of the PVMPROP/CRI Message Format

Part	Length	Notes
PVM/PVMG	4	A 4 character constant; there is a blank character at the end of the character string. This identifies whether the message is from PVM or PVMG.
nnnn	4	This is the PVM or PVMG internal message number
00*ll*	4	The message line number. When the message is encoded, as in PVMPROP messages and CRI responses to a *T.* prefix, the value will always be 0001.
nodeid	8	Contains the node ID of the node where the message originated. If the node ID is less than 8 characters long, the string is filled with blanks on the right.
identifier	6	**For command responses**: the identifier from the original command prefix. If the identifier is less than six characters long, the string is filled with blanks on the right. **For error messages copied to a PVMPROP user ID, or for responses to an *S.* prefix, or for *T.* and *M.* prefixes without using the identifier**: a time stamp in the form *hhmmss*, where *hh* is hours, *mm* is minutes, and *ss* is seconds.
message text	varies	**For messages sent to a PVMPROP user ID or responses to a *T.* prefix**: contains an encoded message. **For responses to an *M.* or *S.* prefix**: contains the actual message text.

Deciphering Encoded Messages

PVM and PVMG send messages to PVMPROP user IDs and *T.* prefix responses in an encoded form using the CP SMSG command. An encoded message contains all of the information needed to know everything the message contains, without the natural language text that a human operator would require. Encoded messages do not contain message information that could change depending on the language that PVM or PVMG are using. This allows application programs that receive encoded messages to be language-independent. Table 26 describes the format of PVM and PVMG encoded messages.

The message will not be encoded if the *S.* or *M.* prefixes are used. *T.* prefixes will cause messages to be encoded unless:

- A CP command response is being echoed by PVM
- The encoded message contains a message from a PVM node not running PVM Version 2 or Version 1.4. The portion of the message passed from the other PVM node will not be encoded.

PVM and PVMG modify an encoded message before sending it so that the encoded message does not contain the *break* character (X'15'). To reverse the modification, your application program should:

- Convert each X'FF' in the encoded message to X'15'.
- Divide the message number by 2, including the embedded message number, if present.

Table 26. Encoded Message Format

Bytes	Meaning
0-1	The PVM or PVMG message number. For PVM or PVMG messages that are issued without a numbered identifier, the message number has a value in the range 7000 through 7999
2-3	The message version number. Some PVM and PVMG messages are broken down into versions in the repository files. This is to allow greater flexibility for different translations. Information sent in an encoded message can vary depending on the message version.
4-6	A three-character module suffix of the PVM or PVMG module issuing the message. These bytes are not used if the message is an unnumbered PVM or PVMG message.
7	The number of substitution variables in the remaining bytes.
8-end	Substitution variables for this message. The format of these variables is: **Byte 1** The length of the substitution variable. **Byte 2** The first byte of the substitution variable. **Note**: There are two special cases for the last variable of an encoded message: • If the length byte is 127 (X'7F'), it indicates that the variable is a 16-byte time stamp in the form "MM/DD/YYHH:MM:SS". • If the length byte is 128 (X'80') or higher, it indicates a PVM or PVMG unnumbered message in encoded form, the length of which is the contents of the length byte minus 128.

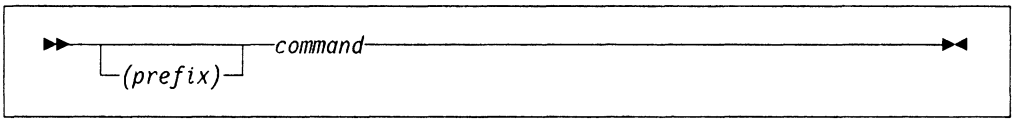
To find out the meanings of the substitution variables used in PVM messages, you can:

- Print and read *Section Two* of the READTHIS FIRST file located on the CMS system extension minidisk (MAINT 19E).
- Look at the on-line message description, if the message has an identifier. Look at the *VM/Pass-Through Facility Messages*, SC24-5648 if the message has no identifier.
- Look at a copy of the source file for the PVM message repository. The file name of the message repository is DVM*langid* REPOS, where *langid* is a language ID. (For example, DVMAMENG REPOS is the source file for the American English message repository.) The message repository can be found on the PVM source staging minidisk (MAINT 39E).

For a description of the parts of a PVM message repository, including message numbers, message versions, and substitution variables, see "Creating and Using a Message Repository" on page 238.

Using Command Prefixes

PVM commands can be preceded by a prefix string as described earlier in this chapter. This prefix tells PVM how to respond to a command. The format for command prefixes is:



Where:

prefix
Is one of the following:

Prefix	Meaning

L.*langid*	Tells PVM or PVMG to send any command responses using the language identified by *langid*. This prefix is intended for use by a general user of PVM or PVMG who needs to see PVM or PVMG command responses in a language other than the default. You can use the QUERY command with the LANGS option to display the language IDs of the languages that are available on PVM. For example, if you issue:
	`sm pvm (l.uceng) q lang`
	you will receive a response similar to:
	`PVM AT RMTPVM1 IS ENABLED FOR LANGUAGE AMENG`
	`PVM AT RMTPVM1 IS ENABLED FOR LANGUAGE UCENG`
	`PVM AT RMTPVM1 IS ENABLED FOR LANGUAGE KANJI`
	`OUR LOCAL DEFAULT LANGUAGE IS AMENG`
	`OUR BACKUP NETWORK LANGUAGE IS UCENG`

T.*identifier*	Tells PVM or PVMG to send the command response in language-independent form using the CP SMSG command. This prefix is intended for use by PVM or PVMG application programs. The *identifier* is a character string that is returned in the command response. The *identifier* can be up to 6 characters long. The T. is optional and can be omitted. For example, (*xxx*) is the equivalent of (T.*xxx*), since T. is the default. Alternately the *identifier* can be omitted. In this case, a time stamp is returned in the command response in place of an identifier. For example, (T.) will return a response in an encoded format. See "The Command Response Interface" on page 252 for more information.

Prefix	Meaning

M.*identifier*	Tells PVM or PVMG to send the command response in the default natural language, prefixed with the 30-byte CRI header (described in "Message Format" on page 253), using the MSGNOH (or MSG) command. The *identifier* is a character string that is returned in the command response. The *identifier* can be up to 6 characters long. This tag is used primarily for testing PVM or PVMG responses for system applications using the command response interface. It can be used by an application when routing a natural language message to a human operator. See "The Command Response Interface" on page 252 for more information.
	
	For example, if you issue:
	`sm pvm (m.tag) q lang`
	
	you will receive a response similar to:
	`PVM 0145 0001 RMTPVM1 TAG PVM at RMTPVM1 is enabled for language AMENG`
	`PVM 0145 0001 RMTPVM1 TAG PVM at RMTPVM1 is enabled for language UCENG`
	`PVM 0145 0001 RMTPVM1 TAG PVM at RMTPVM1 is enabled for language KANJI`
	`PVM 0146 0001 RMTPVM1 TAG Our local default language is AMENG`
	`PVM 0146 0001 RMTPVM1 TAG Our backup network language is UCENG`
	`PVM 0001 0001 RMTPVM1 TAG Your Q command is complete`

S.*langid*	Tells PVM or PVMG to send the command response, prefixed with the 30-byte CRI header (described in "Message Format" on page 253), in the language identified by *langid*, using the MSGNOH (or MSG) command. This tag is used primarily for testing PVM responses for system applications using the command response interface. It can be used by an application when routing a natural language message to a human operator. See "The Command Response Interface" on page 252 for more information.
	
	For example, if you issue
	`sm pvm (s.uceng) q lang`
	
	you will receive an upper case response similar to:
	`PVM 0145 0001 RMTPVM1 124414 PVM AT RMTPVM1 IS ENABLED FOR LANGUAGE AMENG`
	`PVM 0145 0001 RMTPVM1 124414 PVM AT RMTPVM1 IS ENABLED FOR LANGUAGE UCENG`
	`PVM 0145 0001 RMTPVM1 124414 PVM AT RMTPVM1 IS ENABLED FOR LANGUAGE KANJI`
	`PVM 0146 0001 RMTPVM1 124414 OUR LOCAL DEFAULT LANGUAGE IS AMENG`
	`PVM 0146 0001 RMTPVM1 124414 OUR BACKUP NETWORK LANGUAGE IS UCENG`
	`PVM 0001 0001 RMTPVM1 124414 YOUR Q COMMAND IS COMPLETE`

command
is a valid PVM or PVMG command.

For example, if you entered:

```
smsg pvm (l.kanji) query langs
```

PVM sends the response to you in Japanese (provided that the local PVM or PVMG is enabled for Japanese).

Notes:

1. The parentheses shown in the format box are a required part of the command prefix.
2. You cannot use a command prefix:
   - With the PASSTHRU command
   - With the MPVM command
   - With PVM system programmer commands (PVMBLD, PVMCMP, and PVMGEN)

- If you are logged on the PVM or PVMG user ID.

3. To find out if the language you want is available on a PVM or PVMG system, use the QUERY LANGS command. For example, the PVM response to the command:

   ```
   smsg pvm query langs
   ```

   tells you the languages that are available to the local PVM system named pvm.

4. Some languages require special display stations. PVM and PVMG use the requested language without regard to display station requirements.
5. The command prefix is ignored when it is used by the PVM or PVMG operator while logged on he PVM or PVMG user ID.
6. Messages sent to other destinations as the result of your command are not affected by the command prefix.

Sample Application

The PVMSNAP EXEC file on the CMS system extension minidisk is a sample application of the command response interface and PVMPROP. PVMSNAP EXEC traps and processes SMSG messages sent by PVM or PVMG. Although the application is written using the REXX language, it is not an endorsement of one tool over another. You can write your applications using the tools of your choice.

GUPI end

Chapter 16. Application-to-Application Overview

GUPI

The PVM PCCF task allows a CMS virtual machine to request one or more PVM sessions with itself, another virtual machine on the local system, or another CMS virtual machine on any PVM node in the network that contains this facility. Intermediate nodes need not have PCCF support.

To use the PVM PCCF facilities, the application programmer is expected to be familiar with VM IUCV protocols. IUCV information is available in:

- *VM/ESA CMS Application Development Guide for Assembler* (SC24-5452)
- *VM/XA SP: CP Programming Services* (SC23-0370)
- *VM/XA SP: CMS Application Program Development Reference* (SC23-0402).

After the session is established, the virtual machines can communicate asynchronously using the IUCV SEND with the one-way option, or synchronously using the IUCV SEND with reply option.

The data may be of any format and may be character or binary. The data length may be up to 32768 bytes for the SEND message or 32768 bytes for the REPLY message. Any combination of message classes can be used.

PCCF Data Flow

A PVM PCCF session is established by a virtual machine issuing an IUCV CONNECT request to the PVM virtual machine. The PVM target *nodename* is specified in the first 8 bytes of the user data field (IPUSER) of the IUCV CONNECT parameter list, and the *vmid* of the target virtual machine is specified in the second 8 bytes of that field. User virtual machines must have directory authorizations for IUCV communication with the PVM virtual machine, unless the PVM virtual machine has the IUCV ALLOW directory statement. PVM will forward the CONNECT request to the target PVM machine, which will then issue an IUCV CONNECT to the target vmid. The target *vmid* is passed the origin PVM *nodeid* and the origin *vmid* in the user data field of the target PVM's IUCV CONNECT parameter list. If the CONNECT is accepted, the session is completed by PVM, and the original CONNECT is then accepted by the origin PVM. At this point, either user virtual machine can issue IUCV SENDs to pass data back and forth. The CONNECT and ACCEPT options (PRTY, PRMDATA, QUIESCE) are passed from one side to the other.

If the path cannot be completed, a SEVER is done by the origin PVM system.

If both user virtual machines issue a CONNECT for the opposite end, PVM will attempt to start two sessions, and each virtual machine will receive a pending CONNECT request. The virtual machines must decide which path to SEVER and which to ACCEPT.

© Copyright IBM Corp. 1991,1993

Data Transmission

After the session is established, data can be sent between VM applications using an IUCV SEND. If the SEND is one-way, the sending machine will receive the message complete interrupt as soon as the local PVM receives the data, not when the target virtual machine receives the data. The data is then routed to the target node. The sending virtual machine can immediately issue another SEND, thereby causing a stream of data to flow. Because of this, several pieces of data can be transmitted at the same time for the same session. PVM uses end-to-end pacing to limit bottle-necks in this situation. If the number of outstanding transactions reaches a specified limit, the next SEND will not be received by the local PVM virtual machine until the number of outstanding transactions falls below a specified level.

If the SEND requires a REPLY, PVM will send the data to the target node and issue a SEND with REPLY at that node. The reply data is routed back to the sender, and PVM does a REPLY. In this case, the sender will not get the message complete interrupt until the reply is returned.

Any target message class may be used in the IUCV SEND parameter list, and PVM will reflect the same at the target site.

The message size limitations are 32768 bytes for the SEND message and 32768 bytes for the REPLY message, provided both the origin and target PVM nodes are running in VM environments.

The IUCV QUIESCE and RESUME functions will be reflected through to the opposite end. If a IUCV QUIESCE is issued from side A to B, but side B has issued a SEND before the IUCV QUIESCE is reflected, that IUCV SEND will be held at PVM on side A until side A issues an IUCV RESUME.

Although CP queues IUCV PRIORITY messages ahead of non-PRIORITY messages, message transmission within PVM is based upon a First-In-First-Out (FIFO) procedure.

Session Termination

PVM handles both normal and abnormal terminations. These processes are described in this section.

Normal Termination

Either virtual machine may terminate the PVM session by issuing an IUCV SEVER to the path. PVM will then terminate the session and reflect an IUCV SEVER to the other side.

To ensure that any one-way IUCV SENDs that may still be in transit are processed at the opposite end on receipt of a IUCV SEVER from the local application, PVM will transmit a null data packet to the opposite end with a pacing acknowledgement request. It will then wait for all outstanding acknowledgement requests to be satisfied before terminating. At the opposite end, on receipt of a null data packet with a pacing request, the packet will not be processed until all outstanding SENDs have completed. If this side of the path is IUCV quiesced, any queued IUCV SENDs will be purged.

Abnormal Termination

If the session terminates for any reason other than the normal termination, PVM will reflect a SEVER to both sides. When the virtual machine receives an IUCV SEVER interrupt, it must also issue an IUCV SEVER for that path in order to reuse that *pathid*.

PCCF Application-to-Application Restrictions

PCCF application-to-application restrictions are as follows:

- Maximum data length on an IUCV SEND/REPLY is 32768 bytes if both the target and origin CP system is VM/SP Release 4 or higher. Otherwise the maximum length is 4096 bytes.
- CP IUCV PRIORITY cannot be used unless authorized by appropriate CP Directory statements.
- One-way SENDs are not synchronized, and the *message complete* interrupt does **not** mean that the opposite end has received the data.

Application-to-Application Communication Sample Program

The following illustrates the sequence of functions invoked when two virtual machines communicate, using IUCV protocols, through PVM. In this illustration, a virtual machine with vmid USERX is communicating with a virtual machine with vmid USERY. USERX is located at a different PVM node than USERY.

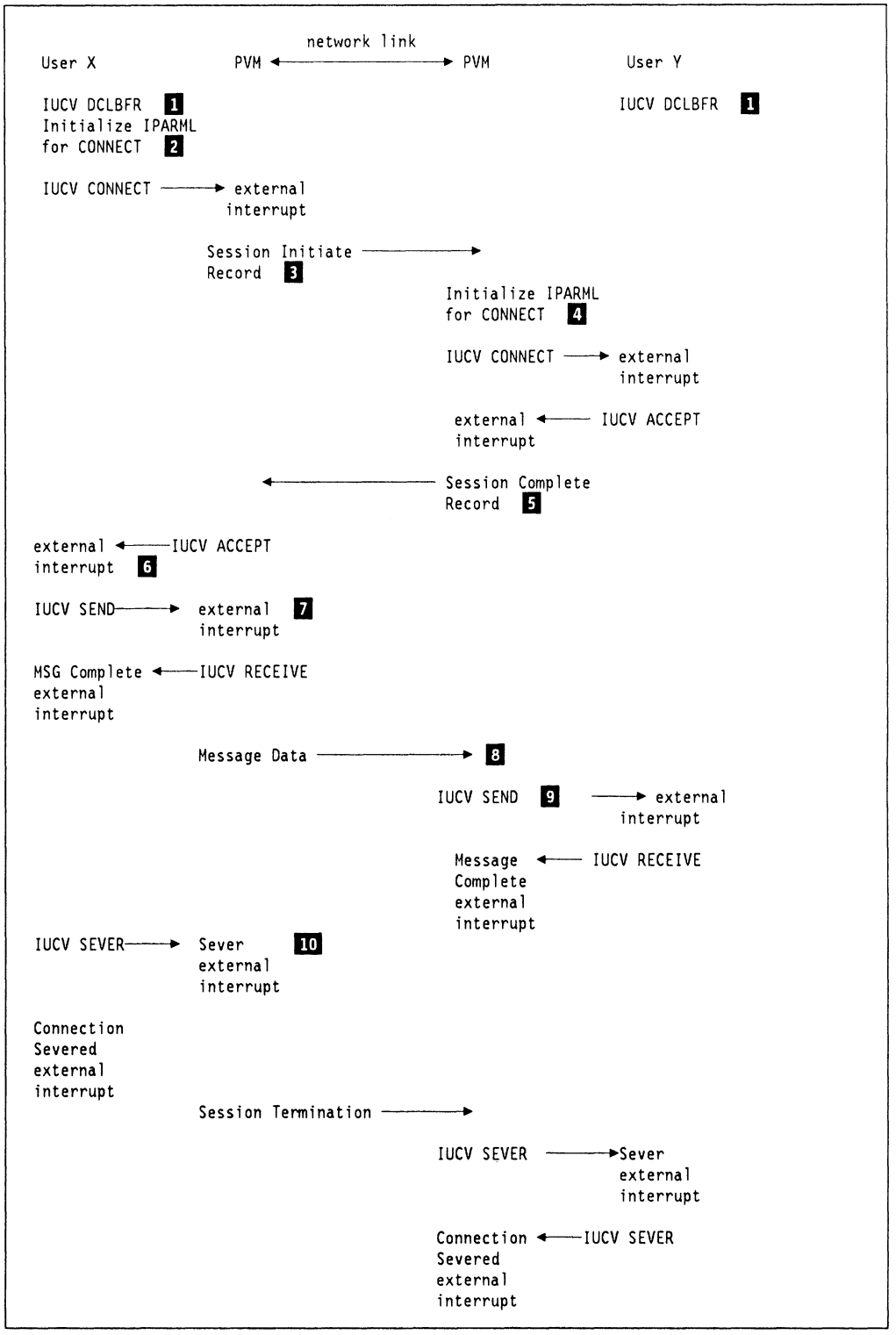


Figure 40. Application-to-Application Communication Sample Program

The IUCV responses which the User X (origin) and User Y (target) virtual machines receive as a result of invoking IUCV commands, reflect the result of communicating with PVM and not with each other. For instance, if the the User X virtual machine connects to the local PVM virtual machine, the return code in the IPARML IPRCODE field is zero regardless of whether the User Y virtual

machine is logged on. If the User Y virtual machine is not logged on, the User X virtual machine is passed an IUCV SEVER.

The flow of the Figure 40 on page 262 is described in greater detail below.

- **1** Both user virtual machines must independently invoke the DECLARE BUFFER function.
- **2** The target nodeid and target vmid are moved into the 16 byte IPUSER field of the IPARML parameter list. The origin PVM vmid is moved into the IPVMID field.
- **3** The origin PVM node has recognized that the USERX virtual machine wants to communicate with the USERY virtual machine at a remote PVM node. It prepares a Session Initiate record and forwards that record to the remote PVM node. The remote PVM node may be on an adjoining link, or it may be several links away. Intermediate PVM nodes do not have to have an active PCCF task.
- **4** The target PVM node has received the Session Initiate record and it prepares an IUCV CONNECT request by initializing the user data field of the IPARML parameter list with the origin PVM nodeid and the origin virtual machine vmid, USERX.

  The IPVMID is set to the target virtual machine vmid, USERY. The IUCV CONNECT request is then invoked.
- **5** The target user virtual machine signals acceptance of the IUCV CONNECT request by invoking an IUCV ACCEPT to the PVM virtual machine. PVM then prepares a Session Complete record and forwards it to the origin PVM node.
- **6** The origin PVM node has received the Session Complete record and informs the origin virtual machine, USERX, by invoking an IUCV ACCEPT.
- **7** The origin and target virtual machines may now send data to each other.

  The target virtual machine issues a one-way IUCV SEND to PVM. PVM performs an IUCV RECEIVE which results in an IUCV Message Complete external interrupt to the origin virtual machine. Because this is a one-way communication the origin virtual machine is now free to send another message to PVM. PVM's pacing support will control the number of messages that the origin can have outstanding before the target virtual machine is able to receive them. The origin virtual machine can also use two-way sends, but it will not get the Message Complete external interrupt.
- **8** The origin PVM node moves the data into PVM message packets and forwards the data to the target PVM node.
- **9** The target PVM node sends the data to the target virtual machine. The Message Complete external interruption indicates that the virtual machine has performed an IUCV RECEIVE. This confirmation is not returned to the origin virtual machine on one-way communications. If the origin virtual machine requires confirmation it should use the IUCV two-way protocol with a reply expected.

- **10** The origin virtual machine has sent all of the data it intends to send and terminates its IUCV session by issuing an IUCV SEVER. PVM will send to the target any remaining messages that it has not completed sending and then terminate the local session and forward a session termination to the remote PVM node to terminate its session. The remote PVM node notifies the target virtual machine of the terminate through an IUCV SEVER.

GUPI end

Chapter 17. Customizing CVIEW

After the CVIEW distribution tape is installed, you can modify the installed options to meet the needs of your establishment. CVIEW provides two product files, the CONSULT IDNAME file and the DXGKEYS LIST file, that you can edit. CVIEW also provides a customization exec, DXGINIT, which can be executed to incorporate your changes. This chapter provides a detailed description of each installed option and instructions on modifying them.

Note: DXGINIT must be run before testing and using CVIEW.

Listed Consultants — CONSULT IDNAME

You can use the CONSULT IDNAME file to create a list of consultants for the GIVE and RECEIVE panels that are displayed to the consultant and recipient during setup. You can edit this file to define up to 11 listed consultants. For each listed consultant, you can specify name, subject or area of responsibility, telephone number, and password.

The CONSULT IDNAME file has a fixed format. Listed consultant information is added to the last 11 data lines and each data line begins with a number from 2 to 12. This number becomes the consultant's unique identifier, known as the **consultant's ID**. The field format of each data line is as follows:

Field Name	Columns	Length
Consultant's name	10 - 33	24
Subject/responsibility	35 - 63	29
Phone	65 - 79	15
Password	81 - 88	8

Defining Listed Consultants

To list consultants for your establishment, do the following:

1. Edit the CONSULT IDNAME file. **Do not** delete unused data lines and **do not** change the numbers at the beginning of each data line.
2. Enter the names of your consultants, their subjects or areas of responsibility, their telephone numbers, and their passwords. The use of passwords is optional. Assigning a consultant a password prevents another user from using the consultant's unique identifier for a consult session. Passwords can be added, deleted, or changed later using the PASSWORD privileged command.
3. After making all the entries, file the updated file. The entries, excluding passwords, are displayed to the consultant and recipient during setup mode to identify the consultant.

You can edit the CONSULT IDNAME file whenever you need to change your list of consultants. To incorporate CONSULT IDNAME file changes, execute the customization exec, DXGINIT, to build new GIVE and RECEIVE panels. **Do not** edit the CONSULT IDNAME file or execute DXGINIT while CVIEW is in operation.

© Copyright IBM Corp. 1991,1993

Program Function Key Assignments — DXGKEYS LIST

Use the DXGKEYS LIST file to assign PF keys for the HELP, QUIT, HOME, and SESSION functions on the CVIEW product panels. You can change only the PF key number, not the file format or function. After installation, the DXGKEYS LIST file contains the following records:

```
HELP = PF1
QUIT = PF3
HOME = PF4
SESSION = PF12
```

PF Key Functions

The **HELP function** displays an explanation panel to the user. The explanation panel contains additional information about the panel being displayed when the user presses the PF key for HELP. The user returns to the previous panel by pressing the ENTER key.

Using the **QUIT function** from a setup panel drops a user out of setup mode (and, thus, out of CVIEW). In addition, if a consultant presses the QUIT PF key from his CONSULT SESSION CONTROL panel, the consult session ends. If a moderator presses the QUIT PF key from his CONFERENCE CONTROL panel or the CONFERENCE PARTICIPANTS panel, the conference session ends. Any user pressing the QUIT PF key when the NOTIFY panel is being displayed terminates the entire session **for all participants**. When the QUIT PF key terminates a session, all session participants except the one who pressed the QUIT PF key return to the HOME panel. The one who pressed QUIT exits CVIEW.

The **HOME function** returns a user to the HOME panel during setup mode. In addition, if a consultant presses the HOME PF key from his CONSULT SESSION CONTROL panel, the consult session ends and the HOME panel is displayed to the recipient also. If a moderator presses the HOME PF key from his CONFERENCE CONTROL panel or the CONFERENCE PARTICIPANTS panel, the conference session ends, and the HOME panel is displayed to all conference participants. Any user pressing the HOME PF key when the NOTIFY panel is being displayed terminates the entire session **for all participants**, and returns all session participants to the HOME panel.

A moderator uses the **SESSION function** to start a conference session after at least one conference participant completes setup. The moderator can also return to a conference session in progress from the CONFERENCE PARTICIPANTS panel by pressing the SESSION PF key.

Changing PF Key Assignments

To change the PF key assignments for your installation, do the following:

1. Edit the DXGKEYS LIST file. **Do not** change the words 'HELP, QUIT, HOME, or SESSION' and **do not** delete any records.
2. Enter 'PF1' through 'PF12' to change the current setting for a function. A PF key must be assigned to each function. The same PF key cannot be assigned to more than one function. Your definitions for PF1 through PF12 will be duplicated for PF13 through PF24.

3. After changing the PF key assignments, file the updated file. When your changes are incorporated, your PF key assignments will appear on the CVIEW panels.

You can edit the DXGKEYS LIST file whenever you need to change PF key assignments. To incorporate DXGKEYS LIST file changes, execute the customization exec, DXGINIT, to build a new PF key area on the CVIEW panels. **Do not** edit the DXGKEYS LIST file or execute DXGINIT while CVIEW is in operation.

The Customizing EXEC — DXGINIT

You use the customization exec, DXGINIT, to change the number of concurrent users, the occurrence of the NOTIFY panel, or the defined privileged users, and to incorporate any changes in the CONSULT IDNAME and DXGKEYS LIST files. DXGINIT issues prompting messages to guide you through the customization process. Each customization prompt includes a default. The first time you execute DXGINIT after installation, the customization defaults are the same as the installed options. You enter new options to customize CVIEW for your use. See "Executing the Customization EXEC" on page 268 for the step-by-step procedure.

Concurrent Users

CVIEW can support a maximum of 30 users at separate display stations who have dialed the CVIEW virtual machine. 5 megabytes are required for the full 30 users. When CVIEW is first installed, the number of concurrent users is set to 30, so the initial customization default is 30. You can change this value to any number from 2 to 29.

The NOTIFY Panel

Because a user's display station could be set up to participate in a consult or conference session without his knowledge, CVIEW provides a mechanism to let a user know that another user is sharing his interactive session. The NOTIFY panel advises a user that someone else is watching his interactive session and solicits a response before continuing the session. The consult or conference session is suspended until all NOTIFY panels displayed for the session receive a positive response. When a user presses the ENTER key from the NOTIFY panel, he is indicating that he wants to continue the session. If any user presses the QUIT or HOME PF keys from the NOTIFY panel, it indicates a negative response to the NOTIFY panel and the entire session end **for all participants**.

The display of the NOTIFY panel during CVIEW sessions is optional. The initial customization default is YES.

The NOTIFY panel can be displayed to all participants in a session or to only the active participants in a session. The initial customization default for this option is ACTIVE. (An *active* session participant is one who has entered data or commands since the last NOTIFY panel was displayed.)

NOTIFY panels appear during a CVIEW session at two different time intervals: so on after the session begins and at regular intervals thereafter. The initial customization defaults for the two intervals are 15 seconds and 9000 seconds. You can change either or both of these values to any number of seconds other than zero.

Privileged Users

Privileged users have the same capabilities from their own virtual machine consoles as an operator logged on to the CVIEW virtual machine console. CVIEW supports a maximum of 15 privileged users. A privileged user can send CVIEW privileged commands to the CVIEW virtual machine for execution from his own virtual machine console. The privileged commands control and administer the CVIEW virtual machine. They are described in "CVIEW Commands" on page 124. You should review these commands before defining privileged users. The initial customization default defines the CVIEW and OPERATOR user IDs as the only privileged users.

Note that if you assign a privilege class of 'B' to the CVIEW virtual machine, any privileged user can send any of the privilege-class-B CP commands to the CVIEW virtual machine for execution. Some privilege-class-B CP commands affect the operation of the CVIEW virtual machine and system resources. In addition, if you assign a privilege class of 'G' to the CVIEW virtual machine, privileged users can enter commands, such as LOGOFF, that affect the operation of the CVIEW virtual machine.

Executing the Customization EXEC

After you edit the CONSULT IDNAME and DXGKEYS LIST files, you can execute DXGINIT. To change a customization default, enter the value you want to use; to accept a customization default, press ENTER in response to the prompting message. To execute DXGINIT, do the following:

1. Log on to the CVIEW virtual machine. (Enter: L CVIEW
2. If the CVIEW virtual machine is in operation, enter: SHUTDOWN
3. Enter: DXGINIT
4. Customize issues the first prompting message:

   Have you entered the consultant IDs in the CONSULT IDNAME file?
   Enter Yes or No Press ENTER for YES.

   If you have edited the file to list your consultant or you do not want to list consultants, press the Enter key and proceed to step 5.

   If you have not edited the file and need to list your consultants, enter *No* and DXGINIT will end. You must then edit the CONSULT IDNAME file and restart this procedure at step 3.
5. Customize issues a prompt requesting the number of concurrent users:

   How many users can DIAL into CVIEW?
   Enter a value between 2 and 30. Press ENTER for 4 users.
6. Customize issues a prompt to determine if NOTIFY panels are to be displayed:

   Do you want to use the Notify Panel?
   Enter Yes or No. Press ENTER for YES.

   If your response was *Yes*, proceed to the next step. If your response was *No*, proceed to step 10.
7. Customize issues a prompt to determine which users are to receive the NOTIFY panel in session mode:

   Do you want to notify all participants in a session or only the active participants in a session?

Enter all or Active. Press ENTER for ACTIVE.

8. Customize issues a prompt to determine when the first NOTIFY panel should be displayed:

   How many seconds should elapse until the Notify Panel is shown the first time?
   Enter the number of seconds. Press ENTER for 15 seconds.

9. Customize issues a prompt to determine when subsequent NOTIFY panels should be displayed:

   How many seconds should elapse between subsequent Notify Panels?
   Enter the number of seconds. Press ENTER for 9000 seconds.

10. Customize issues a prompt to define the privileged users for the CVIEW virtual machine:

    List of userids that can issue privileged commands:
    CVIEW OPERATOR

    Press ENTER to keep the current list of userids.

    To replace the list, enter the new list of userids on one line, separating each userid with a blank and then press ENTER.

11. Customization has issued all the prompts that request information. It saves the values you entered and builds all the product panels to incorporate any changes you made to the CONSULT IDNAME and DXGKEYS LIST files. When customization finishes, a completion message appears.

The values you select during customization are saved in the LASTING GLOBALV file on the CVIEW 191 A-disk. The next time you execute the customization exec, the prompts will show those saved values as the customization defaults.

Part 4. Reference Information

Appendix A. PVM Macros Supported as Programming Interfaces 275
Product-Sensitive Programming Interface Executable Macros 275

Appendix B. Tape Layouts for PVM 2.1 277
Distribution Tape 277
Service Tape 278
National Language Feature Tape 279

Appendix C. PVM Commands 281
* (Comment) 286
AUDIT 286
AUTHORIZ 287
CDBC 289
CMD 290
CMS 290
CP 291
CPQ 292
DEFINE 293
DISC 298
DROP 299
EXEC 301
LMSG 302
LOGMSG 303
MPVM 304
MPVMBLD (PVM 2.1) 307
MSG 308
PASSTHRU 309
PAUSE 313
PMSG 314
PVMBLD 315
PVMCMP 316
PVMGEN (PVM 2.1) 317
QUERY 318
QUIESCE 321
QUIT 323
RESUME 323
ROUTE 324
RUNPVM 325
SELECT 326
SET 326
SHOW 328
SHUTDOWN 329
START 330
STATUS 332
TRACE 334
VARY 337
CP Commands Used with PVM 338
DIAL 338
SMSG 339
PVM/IPCS Support Commands 339
COMBLOCK 340

© Copyright IBM Corp. 1991,1993

DEVQUE 340
HELP 341
IPCSDUMP 341
IPCSSCAN 342
LINKS 343
MAP 344
ROUTES 345
SYSBLOCK 345
TASKQUE 346
USERQUE 346

Appendix D. PVM Configuration File Records 347
* (Comment) 347
AUDIT 347
AUTHORIZ 349
AUTO 349
BUFFERS 350
CLPORT 351
CLUSTER 353
DUMP 353
EXT 354
GROUP 355
I/O 356
LANG 357
LINK 358
LNODE 360
LOCAL 361
MAXCONN 362
MAXIDLE 362
MAXSESS 363
MAXSUSP 363
MAXTDISC 364
MAXUSERS 364
MSGNOH 365
MSGOPTNS 365
OPID 366
PORT 367
PVMPROP 369
RESID 370
ROUTE 370
SELECT 371
SESSTERM 371
STATTIME 372
SUBSET 372
TDISC 373
TIMEOUT 373
XDBC 374

Appendix E. PVMG Commands 375
AUTHORIZ 378
CDBC 380
CMD 381
DISC 382
DROP 382
INIT 383

LMSG . . . 384
LOGMSG . . . 385
MSG . . . 385
PMSG . . . 386
PVMGBLD (PVM 2.1) . . . 387
QUERY . . . 388
QUIESCE . . . 390
RESUME . . . 391
ROUTE . . . 391
RUNPVMG . . . 392
SELECT . . . 393
SHOW . . . 393
SHUTDOWN . . . 394
SPOOL (PVM 2.1.1) . . . 396
START . . . 397
TRACE . . . 398
CP Command Used with PVMG . . . 399
SMSG . . . 399

Appendix F. PVMG Configuration File Records . . . 401
* . . . 401
APPLN . . . 402
AUTHORIZ . . . 403
DUMP . . . 403
LANG . . . 404
LINK . . . 404
LNODE . . . 405
LOCAL . . . 406
LOGMODE . . . 406
LOGONID . . . 407
LU . . . 408
MSGNOH . . . 410
MSGOPTNS . . . 410
OPID . . . 411
PVMPROP . . . 412
ROUTE . . . 412
SELECT . . . 413
SESSDISC . . . 413
SESSTERM . . . 414
XDBC . . . 414

Appendix G. Setting Up an APPC Link . . . 417

Appendix H. PVM Control Block Formats . . . 419

Appendix I. Manually Dialing a Remote 4300 Processor . . . 437

Appendix J. PVM Access Security Exits . . . 439
Installing and Building the Security Exits for PVM 2.1 . . . 440
The REMOTE AUTH File . . . 440
Definitions of the REMOTE AUTH File Entries . . . 442
GATELINK . . . 442
REM-CMD . . . 443
REM+USER . . . 444
SNA-USER . . . 445

VM-USER 446
Messages 448
Format of SECAUDIT LOG Records 449
Controlling Local Users Using the REMOTE AUTH Table 450
Controlling Remote Users by Using the REMOTE AUTH Table 453

Appendix K. CVIEW Product Panels 457
Setup-mode Panels 457
Notify-mode Panels 465
Control-mode Panels 467

Appendix A. PVM Macros Supported as Programming Interfaces

The macros identified in this appendix are provided to allow a customer installation to write programs that use the services of PVM. Only those PVM macros (other than those provided by VM) identified in this appendix, should be used to request or receive services of the PVM component of the VM/Pass-Through Facility.

Product-Sensitive Programming Interface Executable Macros

The following section lists the PVM executable macros in DVMMAC MACLIB that are supported as Product-Sensitive programming interfaces.

- DVMENTR (see "DVMENTR" on page 211 for further details.)
- DVMRETN (see "DVMRETN" on page 211 for further details.)
- DVMXTRN (see "DVMXTRN" on page 212 for further details.)

© Copyright IBM Corp. 1991,1993

Appendix B. Tape Layouts for PVM 2.1

This section describes the tape layouts for the PVM distribution, service, and national language feature tapes.

Distribution Tape

PVM consists of twelve files shipped on a distribution tape. The procedures used to install the PVM distribution tape are discussed in "Loading Files - The Quick Method (Using INSTFPP)" on page 48.

Tape File 1
PVM Identifier File, PVM Installation EXEC, softcopy books, and PVM PRODPART (I5684100), needed for VM/ESA installation, loaded to MAINT's 191 disk.

Tape File 2
Memo-to-users (I5684100 MEMO) loaded to MAINT's 191 disk.

Tape File 3
Sample PVM and PVMG directory files loaded to MAINT's 191 disk.

Tape File 4
PVM system file (PVM module, config files and loadlib) loaded to MAINT's 36E disk (PVM's 191 disk).

Tape File 5
System extension files (DVMUSI module, MPVM and PASSTHRU commands) loaded to MAINT's 19E disk.

Tape File 6
Selected product source files (Assemble, copy , and macros) loaded to MAINT's 39E disk.

Tape File 7
Product base text files (Text, Maclib, Cntrl, and service/build execs) loaded to MAINT's 49E disk.

Tape File 8
PVM HELP files optionally loaded to MAINT's 19D disk.

Tape File 9
IPCS help and text files loaded to MAINT's 193 disk.

Tape File 10
PVMG system files and PVMG LOADLIB loaded to MAINT's 38E disk (PVMG's 191 disk).

Tape File 11
MPVM system files (MPVM user module and macros) loaded to MAINT's 38F disk (PVM's 199 disk).

Tape File 12
Sample VTAM definition file loaded to MAINT's 298 disk.

© Copyright IBM Corp. 1991,1993

Service Tape

If your installation receives PUTS from the IBM Support Center, you receive a service tape containing all the data, EXECs, and service tools you need to make updates. The service tape you receive will have 13 logical tape files. The procedures used to install the service tape are discussed in Chapter 11, "Applying Service to PVM 2.1" on page 141.

Tape File 1
5684100A 021013 (product level file) and the 5684100A EXEC (service exec). Load to MAINT's 191 disk.

Tape File 2
5684100A MEMO (memo-to-users file). Load to MAINT's 191 disk

Tape File 3
New or changed SAMPLE files for PVM and PVMG. These are mandatory files. Load to MAINT's 191 disk.

Tape File 4
New or changed PVM SYSTEM DISK files. Contains the PVM MODULE. These are mandatory files. Load to MAINT's 36E/PVM 191 disk.

Tape File 5
New or changed PVM SYSTEM EXTENSION files. Includes DVMUSI MODULE. These are mandatory files. Load to MAINT's 19E disk.

Tape File 6
New or changed PVM SOURCE files. Includes COPY, MACRO, and ASSEMBLE files. These files are optionally loaded to MAINT's 39E disk.

Tape File 7
PVM SERVICE files. Includes AUX and UPDATE files. Optionally load to MAINT's 29E disk.

Tape File 8
New or changed PVM BASE files. Includes CNTRL, EXECS, TEXT, and and MACLIB files. Optionally load to MAINT's 49E disk.

Tape File 9
New or changed PVM HELP files. Includes HELPPVM files. Optionally load to MAINT's 19D disk.

Tape File 10
New or changed PVM IPCS files. Includes TEXT and HELP files. Optionally load to MAINT's 193 disk.

Tape File 11
New or changed PVMG SYSTEM files. Includes PVMG LOADLIB file. Optionally load to MAINT's 38E/PVMG 191 disk.

Tape File 12
New or changed MPVM files. Includes the MPVM user module. Optionally load to MAINT's 38F/PVM 199 disk.

Tape File 13
New or changed VTAM files. Includes SNAMODET ASSEMBLE file. Optionally load to MAINT's 298 disk.

National Language Feature Tape

The national language feature tape consists of 11 tape files. The procedures used to install the national language feature are discussed in Chapter 14, "National Languages on Your PVM System" on page 227.

Tape File 1
The product level file (I5684NLS 021011NL) and the NLS service install exec (I5684NLS EXEC).

Tape File 2
Memo-to-users (I5684100 MEMONL).

Tape File 3
PVM system files. Load to MAINT's 36E (PVM's 191).

Tape File 4
System extension files. Load to MAINT's 19E.

Tape File 5
Message repository source files. Load to MAINT's 39E.

Tape File 6
Message repository object files. Load to MAINT's 49E.

Tape File 7
PVMG system files. Optionally load to MAINT's 38E.

Tape File 8
MPVM system files. Optionally load to MAINT's 38F.

Tape File 9
PVM help files for KANJI. Optionally load to MAINT's 401.

Tape File 10
PVM help files for upper case English. Optionally load to MAINT's 402.

Tape File 11
PVM help files for GERMAN. Optionally load to MAINT's 405.

Appendix C. PVM Commands

This appendix describes the commands and execs that are used to control and use PVM. (In this appendix, execs can be considered synonymous with commands.) It includes discussions (starting on page 338) of two CP commands: DIAL and SMSG. Descriptions of additional commands used with the optional PVM/IPCS support are also included.

Notes on Using Commands

Certain PVM commands can be issued by users identified in AUTHORIZ records in the PVM configuration file. These commands are the *A* commands listed in Table 28 on page 284. All PVM commands are listed in this table. Some commands must be issued by a PVM operator logged onto the PVM user ID. These commands are the *O* commands. A virtual machine user may use any of the unrestricted commands. These commands are the *G* commands. Some commands are intended to be used by the PVM system programmer. These commands are the *S* commands.

Some PVM command options have restriction levels that are different from the remaining options for that command.

If you are not working at the PVM user ID, each PVM command expression must be included in a CP SMSG (Special Message) command that contains the PVM user ID. This sends the command to PVM to be executed. The exception to this rule is when a local installation has provided an exec for each command that automatically puts the *SMSG pvmid* characters in front of the PVM command expression.

You can use a command prefix when issuing PVM commands. Command prefixes tell PVM to send command responses using a particular format. See "Using Command Prefixes" on page 255 for more information about command prefixes. For information on how to use command prefixes in application programming, see Chapter 15, "Facilities for Automating PVM and PVMG Operations" on page 251.

How to Read the Syntax Diagrams

Throughout this book, syntax is described using the structure defined below.

- Read the syntax diagrams from left to right, from top to bottom, following the path of the line.

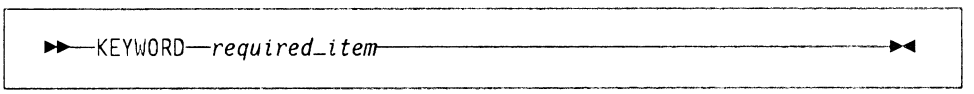
  The ▶▶— symbol indicates the beginning of a command.

  The —▶ symbol indicates that the command syntax is continued on the next line.

  The ▶— symbol indicates that a command is continued from the previous line.

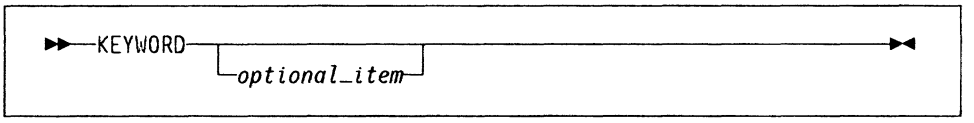
  The —▶◀ symbol indicates the end of a command.

- Required items appear on the horizontal line (the main path).



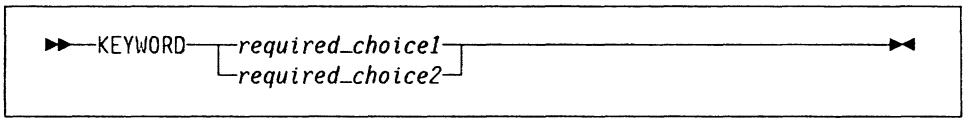
© Copyright IBM Corp. 1991,1993

- Optional items appear below the main path.

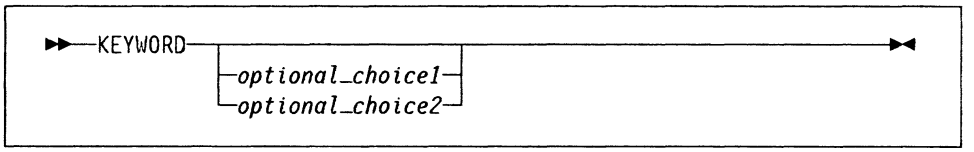


- If you can choose from two or more items, they appear vertically, in a stack.

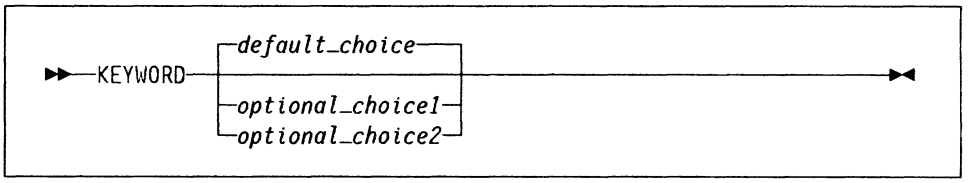
  If you *must* choose one of the items, one item of the stack appears on the main path.



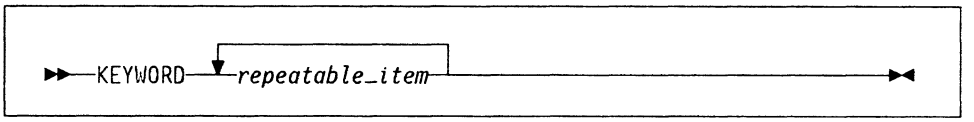
  If choosing one of the items is optional, the entire stack appears below the main path.



  If one of the choices is a default, it appears above the main path.

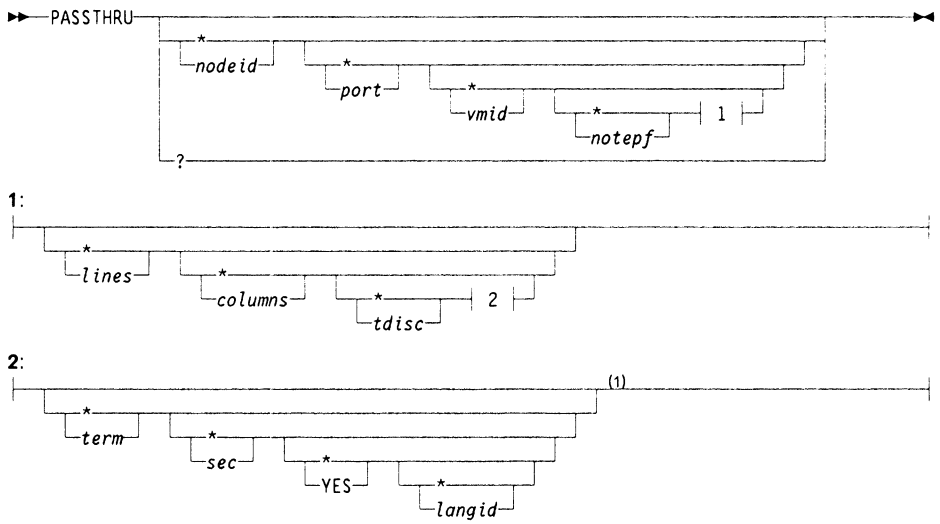


- An arrow returning to the left above the main line indicates an item that can be repeated.



  A repeat arrow above a stack indicates that you can repeat the items in the stack.

The following example of the PASSTHRU command illustrates many aspects of a syntax diagram. An explanation of the PASSTHRU command is provided within the syntax diagram.



Note:

1 PASSTHRU is a positional command. If you wish to enter an option that follows other options that you do not wish to use, you must enter an asterisk (*) for each unused option that occurs before the option you wish to use. For example, if you wish to enter PASSTHRU with the *langid* operand, you must enter an asterisk as a placeholder for each preceding option. Your command would be PASSTHRU * * * * * * * * * *
langid. The ? operand cannot be followed by any other operand, since PASSTHRU ? provides help information about the PASSTHRU command.

Fragments *1* and *2* refer you to a continuation of the positional, sequential, operands. When you have completed entering operands, you can follow the syntax diagram to the end of the command, bypassing any remaining optional operands.

Online HELP is available

As a VM user, you can receive information about selected PVM commands online using the VM/ESA HELP Facility. To display a menu of selected PVM commands, from CMS enter:

```
HELP PVM MENU
```

To display information for a specific PVM command, PASSTHRU in this example, from CMS enter:

```
HELP PVM PASSTHRU
```

If you are not familiar with the VM/ESA HELP Facility, enter `help` to display the main HELP menu or enter `help cms help` for detailed information about the HELP command.

Help for PVM messages with identifiers is only available through the VM/ESA HELP Facility. Only messages without identifiers, PVM IPCS support messages, and Compiler messages are documented in the PVM hardcopy library.

See the *VM/ESA CMS User's Guide, SC24-5460* for a description of HELP and how to tailor HELP. See the *VM/ESA CMS Command Reference, SC24-5461* for the format and syntax of HELP and related commands.

Table 28 lists the commands, their functions, and the users authorized to use them. Note the following about the first column of this table:

Table 27. Key to authorized users.

User	Meaning
A	Alternate operator specified on an AUTHORIZ record in the PVM configuration file.
O	PVM operator logged onto the PVM user ID.
G	General VM user.
S	PVM system programmer.

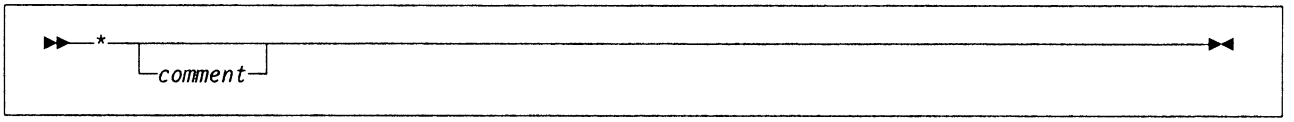
Table 28 (Page 1 of 2). PVM Commands and Execs and Their Functions

User	Command	Function
A O G	*	Comment following asterisk prints out on the PVM operator's display station, but no function is performed. If a PVM audit file is being kept, the comment is also written to that file.
A O	AUDIT	Controls the creation and recording of the PVM audit file.
A O	AUTHORIZ	Adds or deletes entries from the PVM authorized user list. This command must be issued on the local system.
A O	CDBC	Specifies whether display unit can or cannot handle messages in the default language without regard to double-byte character restrictions.
A O G	CMD	Sends PVM commands to another PVM for execution.
A O	CMS	Executes CMS subset commands within PVM.
A O	CP	Executes CP commands within PVM.
A O G	CPQ	Issues one out of a specific set of CP commands on PVM.
A O	DEFINE	Adds or modifies a PVM link, defines a new line group, or adds links to a line group.
O	DISC	Disconnects the PVM user ID.
A O	DROP	Terminates an attached display or printer support task, a line, or a user session.
A O	EXEC	Executes the PVM commands in a specified CMS file.
A O	LMSG	Sends a message to PVM users on a specified link.
A O	LOGMSG	Defines the LOGMSG area on the PVM Selection Menu and on the Initial Menu.

Table 28 (Page 2 of 2). PVM Commands and Execs and Their Functions

User	Command	Function
A G	MPVM	Starts a MPVM session from a CMS user ID.
S	MPVMBLD	Regenerates the MPVM user module.
A O G	MSG	Causes PVM to forward a message to a specified VM user.
A G	PASSTHRU	Starts a PVM session from a CMS user ID.
A O	PAUSE	Causes the PVM command processor to enter a wait state for one second.
A O G	PMSG	Causes PVM to forward a message to specified PVM users. The ALL option is restricted to the *A* and *O* user groups.
S	PVMBLD	Regenerates the PVM MODULE.
S	PVMCMP	Compiles a PVM message repository.
S	PVMGEN	Generates the PVM module that executes in a user's CMS virtual machine.
A O G	QUERY	Displays information about PVM. The AUTHORIZ option is restricted to the *A* and *O* user groups and to the local VM system.
A O	QUIESCE	Quiesces an attached printer or display support task, a line driver task, or the PVM system.
O	QUIT	Terminates the PVM system immediately.
A O	RESUME	Reverses the effect of the QUIESCE command.
A O	ROUTE	Adds, deletes, or changes a node's routing.
O	RUNPVM	Loads and starts PVM.
A O	SELECT	Adds, deletes, or changes nodes assigned to PF keys on the PVM Selection Menu or on the Initial Menu.
A O	SET	Sets the maximum times for idle, suspended, and temporarily disconnected MPVM users, and for the number of users and sessions allowed for MPVM users.
A O G	SHOW	Shows the path of a particular session.
O	SHUTDOWN	Terminates the PVM system.
A O	START	Starts a line driver task, an attached display support task, or an attached printer support task.
A O	STATUS	Displays the status if the system pools or the status of tasks within the PVM system.
A O	TRACE	Initiates or stops the line tracing facility, the PVM multitasking supervisor trace facility, or the tracing of a PCCF support task.
A O	VARY	Alters the availability status of links and devices.

* (Comment)



Purpose

Use the * command to enter comments. This command is non-operational.

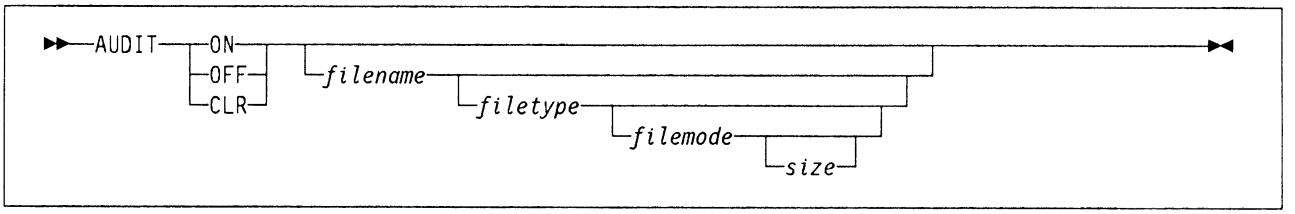
Operands

comment
is the information reflected on the PVM console and in the optional AUDIT file.

Usage Notes

1. When issued from another virtual machine using SMSG, you can use a command prefix with this command to tell PVM how to respond to the command. See"Using Command Prefixes" on page 255 for more information about command prefixes.

AUDIT



Purpose

Use the AUDIT command to dynamically control the creation and recording of the PVM audit file.

Operands

OFF
closes the audit file and stops the recording of the audit file.

ON
opens the audit file. If the audit file does not exist, the AUDIT ON command creates and initializes the audit file.

CLR
erases the audit file, if it exists. Then, AUDIT CLR creates and initializes a new audit file.

filename
is the name of the audit file. If you specify =, *filename* is the current file name. CONSOLE LOG A6 is the default file name, file type, and file mode for the audit file.

filetype
is the file type of the audit file. If you specify =, *filetype* is the current file type.

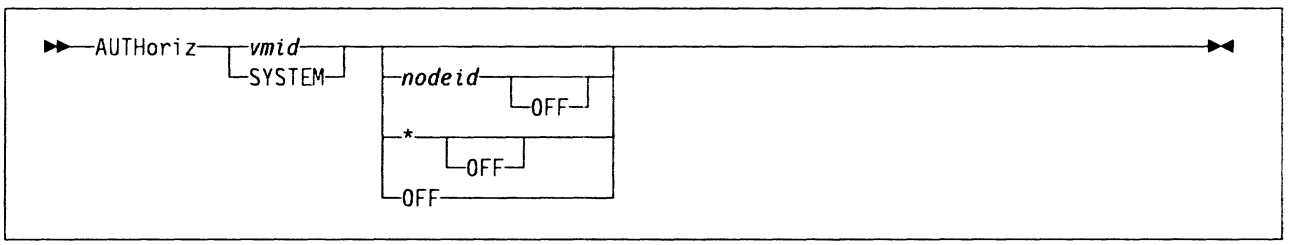
filemode
is the file mode of the audit file. If you specify =, *filemode* is the current file mode. If only the letter is specified, 6 is the file mode.

size
is the maximum size of the file in 1000 records. This value can be from 1 to 1000. For example, if you specify 3, the maximum size of the file is 3000; if you specify 40, the maximum size of the file is 40,000.

Usage Notes

1. This command is restricted to the PVM operator or an alternate operator specified on an AUTHORIZ record in the PVM configuration file.
2. If you specify ON, the specified file is cleared if:
   - The *size* value is specified and is greater than the current *size* value
   - The file exists and is in the wrong format, or record one does not contain a valid pointer.
3. When the audit file is full, the data wraps to the beginning of the file overlaying the existing records.
4. When issued from another virtual machine using SMSG, you can use a command prefix with this command to tell PVM how to respond to the command. See "Using Command Prefixes" on page 255 for more information about command prefixes.

AUTHORIZ



Purpose

Use the AUTHORIZ command to add and delete entries from the PVM authorized user list. When PVM initializes, the authorization list holds up to 30 extra entries. The change is effective only during the execution time of the PVM program. To make this change permanent, you must add or delete an AUTHORIZ record in the PVM configuration file.

Operands

vmid
is the user identification of the CP virtual machine you want to add to, or delete from, the authorization list. If you do not specify * or *nodeid*, this authorization is valid for the local node only.

SYSTEM
authorizes the PVM operator to issue commands to nodes (* or *nodeid*) other than the local node. Use this operand rather than the PVM *vmid*. Without specifying * or *nodeid*, this authorization is for the local node only.

nodeid
is the specific node where *vmid* is authorized. You can specify the same *vmid* from different *nodeid*s.

*
indicates that *vmid* is authorized on any node. Consider this option carefully, because it may be a security risk in a network where user IDs could be duplicated. You cannot add *vmid* to the authorization list using a global node ID (*) and a specific node ID.

OFF
removes the specified user from the authorization list.

Usage Notes

1. This command is restricted to the PVM operator or an alternate operator specified on an AUTHORIZ record in the PVM configuration file and must be issued on the local system.
2. If the operator makes a typographic error in issuing the AUTHORIZ command, PVM may create an entry in its AUTHORIZ table using this incorrect information. If the operator recognizes this error, delete the entry using the **OFF** option of the AUTHORIZ command and submit the correct AUTHORIZ command again.
3. Use the QUERY AUTHORIZ command to display the contents of the AUTHORIZ table.
4. When issued from another virtual machine using SMSG, you can use a command prefix with this command to tell PVM how to respond to the command. See "Using Command Prefixes" on page 255 for more information about command prefixes.

Examples

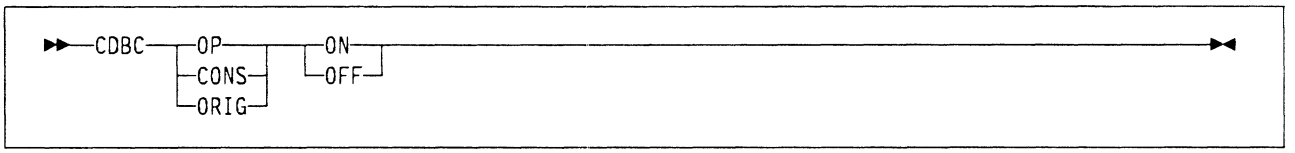
To add authorization for a user ID named JONES to the PVM node NEWYORK, enter the command:

```
auth jones newyork
```

To remove authorization for a user ID named SMITH from all PVM nodes, specify:

```
auth smith * off
```

CDBC



Purpose

Use the CDBC (*C*hange *D*ouble-*B*yte-*C*haracters) command to change the DBCS-exclusions settings for OP, CONS, and ORIG without having to initialize PVM again. These settings are ignored when a user (other than the PVM console operator) specifically requests a language by using a command prefix.

Operands

OP
identifies the CP operator.

CONS
identifies the PVM console.

ORIG
identifies all virtual machine users on the local VM system, other than the CP operator and the PVM operator. That is, ORIG identifies the originators of actions or commands to which PVM messages are sent in response.

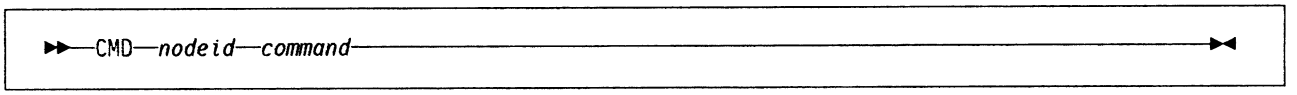
ON
indicates that the default language messages should be issued to the user without regard to DBCS content.

OFF
indicates that the backup language should be used for any messages to the user whenever the default language message that would otherwise be used contains DBCS.

Usage Notes

1. This command is restricted to the PVM operator or an alternate operator specified on an AUTHORIZ record in the PVM configuration file.
2. Issue the CDBC CONS OFF command when you are logged on to the PVM user ID to prevent errors. You can use CDBC CONS ON when you are disconnected.
3. When issued from another virtual machine using SMSG, you can use a command prefix with this command to tell PVM how to respond to the command. See "Using Command Prefixes" on page 255 for more information about command prefixes.

CMD



Purpose

The CMD command provides the user the ability to send a command to be executed at another PVM or PVMG node in the PVM network. You must be authorized at the remote node to have restricted commands issued at that node. The originator receives the output generated from this command. The CMD command is useful for various query commands.

Operands

nodeid
is the node on the PVM network where the command is to execute. This may be a PVM or PVMG node.

command
is any PVM or PVMG command you are authorized to issue.

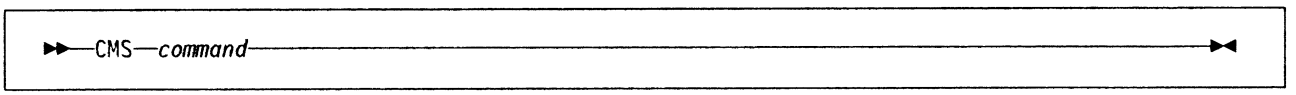
Usage Notes

1. When issued from another virtual machine using SMSG, you can use a command prefix with this command to tell PVM how to respond to the command. Place the prefix in front of the CMD command, rather than in front of the *command* portion of the command. For example, use the command:

   ```
   smsg pvm (1.kanji) cmd rmtpvm5 query system
   ```

   See"Using Command Prefixes" on page 255 for more information about command prefixes.

CMS



Purpose

Use the CMS command to execute CMS commands from within PVM. The output from this command appears only at the PVM user ID display station. Only CMS subset commands are executed using this command.

Operands

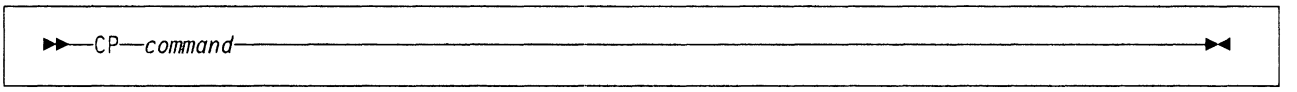
command
is the CMS command executed within PVM.

Usage Notes

1. This command is restricted to the PVM operator or an alternate operator specified on an AUTHORIZ record in the PVM configuration file.
2. When issued from another virtual machine using SMSG, you can use a command prefix with this command to tell PVM how to respond to the command. See "Using Command Prefixes" on page 255 for more information about command prefixes.

Warning: PVM stops functioning while the CMS command is executing. If this interval is long, the communications links and PVM users are affected.

CP



Purpose

Use the CP command to execute CP commands within PVM.

Operands

command
is the CP command executed within PVM. The originator receives the CP output generated from *command*. A maximum of 4096 characters, less the size of the PVM buffer header, of output are returned; the rest are ignored.

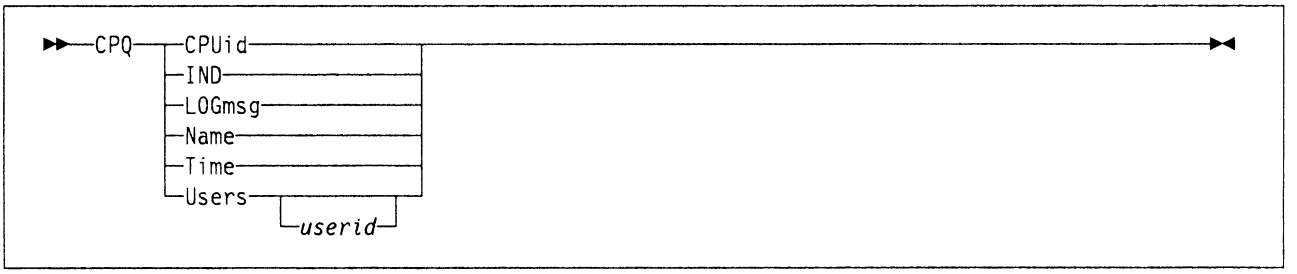
Usage Notes

1. This command is restricted to the PVM operator or an alternate operator specified on an AUTHORIZ record in the PVM configuration file.
2. If you log on PVM, disconnect while SET RUN is OFF, then reconnect, you will not see any log messages or a RECONNECTED message. You will only see the CP READ status. The messages will appear after you issue CP BEGIN.

   However, if you disconnect while SET RUN is ON, then reconnect, you will receive the log messages and a RECONNECTED message.
3. When issued from another virtual machine using SMSG, you can use a command prefix with this command to tell PVM how to respond to the command. See "Using Command Prefixes" on page 255 for more information about command prefixes.

Warning: The system does not check to see if you enter the CP LOGOFF command. If you issue the CP LOGOFF command, you log off PVM.

CPQ



Purpose

Use the CPQ command to issue a specific set of CP commands on PVM.

Operands

CPUid
displays the local node name, the processor model number, and the level of the VM system.

IND
displays the processor use, processor contention, main storage use, and main storage contention.

LOGmsg
displays the current VM log message.

Name
lists all the users logged on and the display station device address associated with each user. If a user is disconnected, DSC is displayed instead of the address.

Time
returns the equivalent of the CP QUERY TIME command executed on the PVM virtual machine.

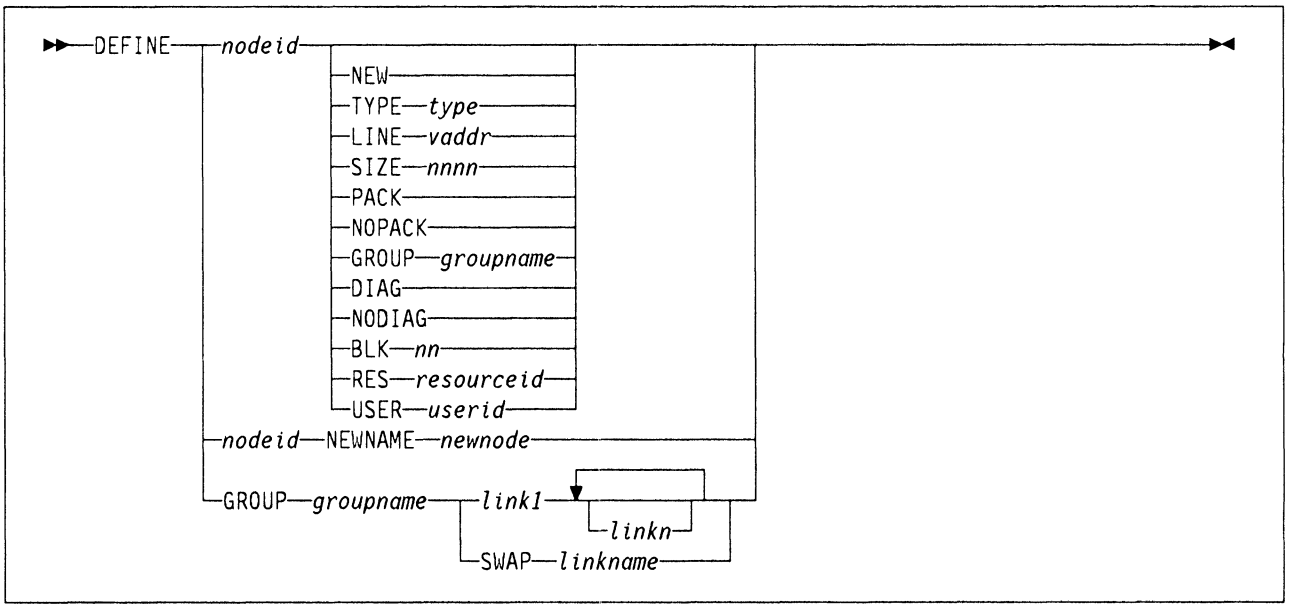
Users
displays the number of users logged on and the number of users logically connected to other virtual machines.

Users *userid*
displays the user identification and the display station device address of the specified user if that user is logged on.

Usage Notes

1. When issued from another virtual machine using SMSG, you can use a command prefix with this command to tell PVM how to respond to the command. See"Using Command Prefixes" on page 255 for more information about command prefixes.

DEFINE



Purpose

The DEFINE command allows a system support person to dynamically add or modify a:

- PVM link
- PVM link to a PVMG virtual machine using an IUCV connection
- PVM link to a remote PVM system using an ESCON channel using a 3088 line driver (see Usage Note 1)
- New line group
- Link to a line group.

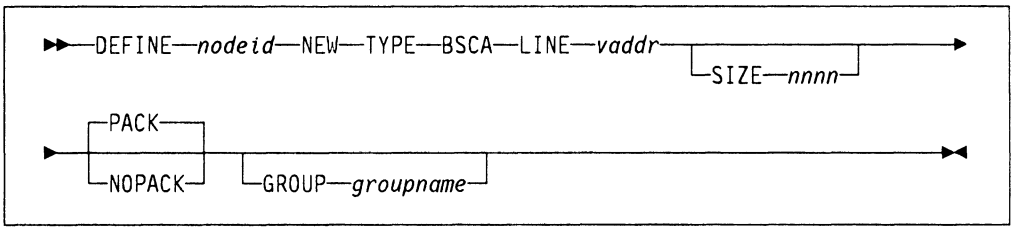
You can add only link types BSCA, CTCA, APPC, and 3088 to a line group. The line address must be compatible with the link type or you receive an error message. You can change only types BSCA, CTCA, APPC, and 3088; and they can only be changed to BSCA, CTCA, APPC, and 3088. However, you can change other options on the other link types, provided the link type supports that option.

Except for SIZE and GROUP SWAP, use the DEFINE command to modify an existing link only if the link is in the DOWN state. If the link is not in the DOWN state, you receive an error message.

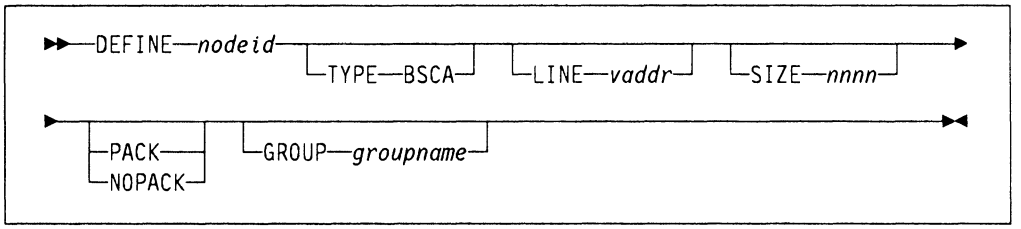
The following are examples to illustrate the format of the line driver types for the DEFINE command. They include formats of the DEFINE command for the specified type of links--BSCA, CTCA, APPC, IUCV, and 3088.

BSCA Format

Use the following format of the DEFINE command to define a new BSCA link.

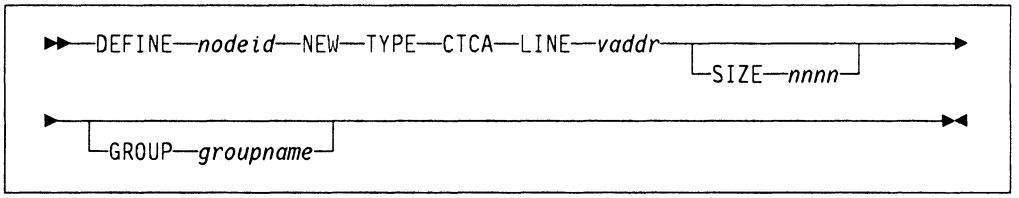


Use the following format of the DEFINE command to change a BSCA link.

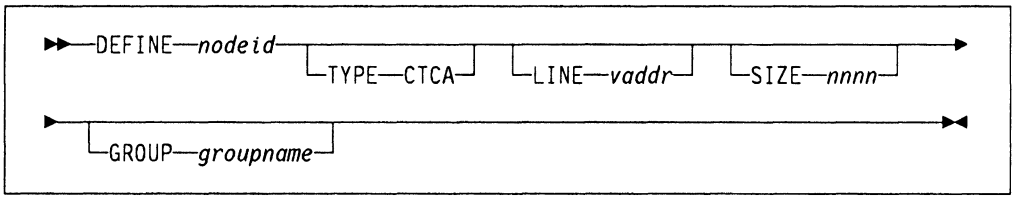


CTCA Format

Use the following format of the DEFINE command to define a new CTCA link.

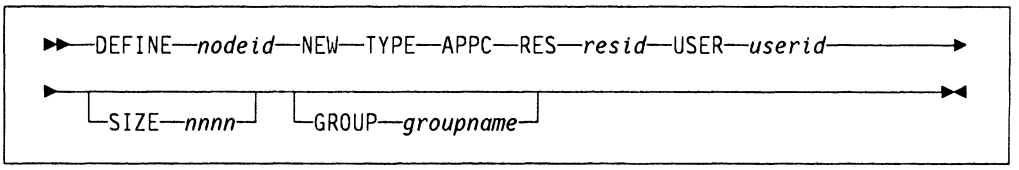


Use the following format of the DEFINE command to change a CTCA link.

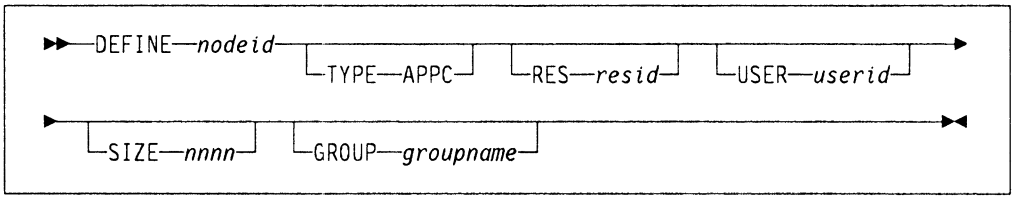


APPC Format

Use the following format of the DEFINE command to define a new APPC link.

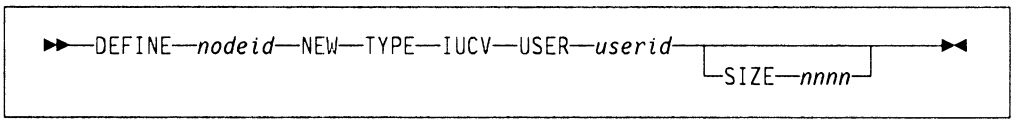


Use the following format of the DEFINE command to change a APPC link.

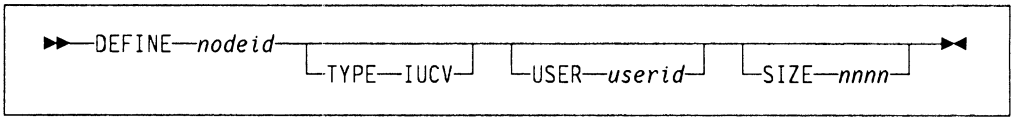


IUCV Format

Use the following format of the DEFINE command to define a new IUCV link.

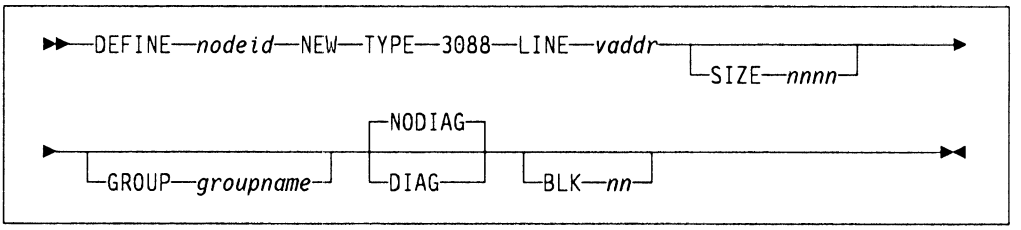


Use the following format of the DEFINE command to change an IUCV link.

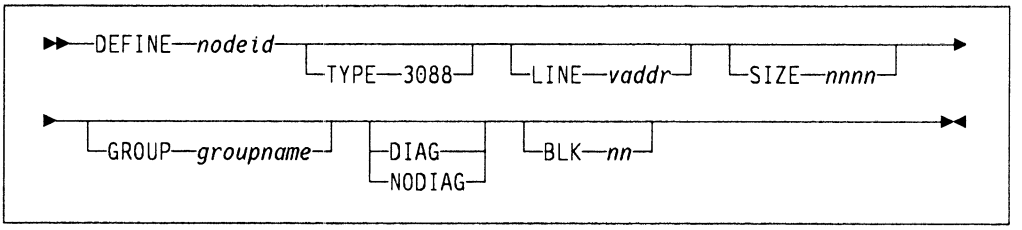


3088 Format

Use the following format of the DEFINE command to define a new 3088 link.



Use the following format of the DEFINE command to change a 3088 link.



Operands

nodeid
is the link node name. The *nodeid* must already exist if you do not specify NEW, or *nodeid* must not exist if you specify NEW. The node name cannot already be used as a GROUP name.

NEW
adds a new link. It must immediately follow the *nodeid*, if present.

TYPE *type*
assigns a new link type to the link. The type must be BSCA, CTCA, APPC, IUCV, or 3088. If you specify NEW, TYPE must be included and must follow the keyword NEW. If you do not specify NEW, TYPE must follow the *nodeid*.

LINE *vaddr*
assigns a new line address to the link. If you specify NEW, LINE must be included. The line address is checked to determine if it already exists for another link.

SIZE *nnnn*
assigns a new block threshold value to the link. This value governs the minimum block size to be transmitted when there is a queue of data to be sent. The value must be between 200 and 4096. If you specify NEW and not SIZE, the default is 4096 if the link type is CTCA, APPC, IUCV, or 3088 and 1000 if the link type is BSCA. This option is not valid for R3270, ROCF, and 327X type links.

PACK
indicates that outbound data compression is used on the link. PACK can only be specified for a BSCA link type. PACK is the default option if this is a new definition and you do not specify either PACK or NOPACK.

NOPACK
indicates that outbound data compression is not to be used on the link. NOPACK can only be specified for a BSCA link type.

GROUP *groupname*
indicates that *nodeid* is part of the group *groupname*. The specified group must already be defined in PVM. The GROUP operand is only valid for link types that support line groups.

DIAG
indicates that DIAGNOSE CODE X'98' REAL I/O is used for this link. DIAG is only valid if the link type is 3088. For this operand to be enabled, the VM environment must support this function and the PVM directory must specify OPTION DIAG98.

NODIAG
indicates that DIAG 98 REAL I/O is not used for this link. NODIAG is only valid if the link type is 3088. NODIAG is the default option if this is a new definition and you do not specify either DIAG or NODIAG.

BLK *nn*
indicates the number of 4K areas tied together for the output transmission buffer. *nn* must be between 2 and 16. The default value is 3. BLK is only valid if the link type is 3088.

RES *resourceid*
indicates the nickname of the resource to which you are connecting. This operand only applies to an APPC type link. The :nickname. field in the CMS communications directory (UCOMDIR NAMES) matches this *resourceid*.

USER *userid*
For APPC: identifies the local user that is requesting a connection to a remote PVM node over an APPC link.

The *userid* (and password) must match the name defined in the :userid. field of the CMS communications directory.

For IUCV: indicates the user ID of the connecting PVMG virtual machine.

NEWNAME *newnode*
modifies the node name of a previously defined link. The restrictions are a:

- Link cannot be currently part of a line group
- Link cannot be started
- New node name cannot be already used as either a link name or a group name.

If any nodes are currently routed to the old link node name, their routing is changed to the new node name of the link.

GROUP *groupname link1 link2 ...*
defines a new line group or adds new links to an existing group. A maximum of eight groups can be dynamically defined.

The *groupname* is the name of the line group. If *groupname* already exists, the link names are added to the current group.

The *link1* is the name of a link assigned to the group. The link must already be defined and cannot be started.

The *link2* is the name of a link assigned to the group. The link must already be defined and cannot be started.

Note: You can only specify line types that may participate as part of a line group. Only one APPC type link can be in each group.

GROUP *groupname* **SWAP** *newlinkname*
takes a link that is not part of a line group, makes the name of this link the name of the group you are defining, and renames this link. The *groupname* that was the name of the original link becomes the name of the newly defined group. The *newlinkname* is the new name you specified for the original link. The *newlinkname* must not already be used as a link name or a group name.

You can execute this function no matter what state the link is in, but it can only be executed for links that support line groups. The associated link need not be dropped for the SWAP function.

Note: Only link types that may participate as part of a line group can be specified.

Usage Notes

1. When defining an ESCON channel line driver to a remote PVM system, the same parameters are accepted as would be used to define a 3088 line driver.
2. Line driver grouping is not permitted for a type of IUCV.
3. This command is restricted to the PVM operator or an alternate operator specified on an AUTHORIZ record in the PVM configuration file.
4. The SIZE option can be used to prioritize links in a group. Sessions go through the link in a group that has the highest size (priority), increasing performance. Links that have lesser size values are used for backup purposes.
5. To dynamically define a new line group and links:
   - Define the links without specifying the GROUP parameter

- Define the GROUP listing the predefined links.

6. To convert a link to a line group:
   - Use the GROUP SWAP operand to convert the link to a line group
   - Define any more links adding the GROUP option.
7. When issued from another virtual machine using SMSG, you can use a command prefix with this command to tell PVM how to respond to the command. See "Using Command Prefixes" on page 255 for more information about command prefixes.

Examples

Suppose you have a link from a PVM system, whose node ID is PVMA, to another PVM system, whose node ID is PVMB. The name of this link is PVMB. If you want to add another link from PVMA to PVMB, you can do so by using the DEFINE command. First you need to define a group. Issue the following DEFINE command:

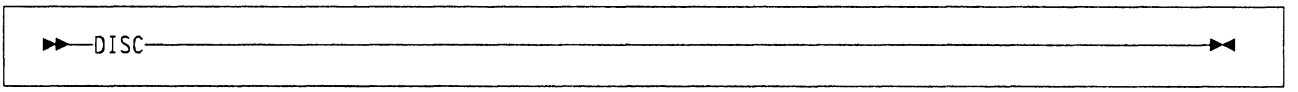
```
define group pvmb swap pvmb1
```

This command performs two functions. It takes the original link name, PVMB, and makes it the group name. Then, the DEFINE command renames the original link name to PVMB1. Therefore, you have a group with the name PVMB and a line in that group with the name PVMB1. You can now add other lines to group PVMB by using the following DEFINE command:

```
define group pvmb link1 ... linkn
```

where `link1 ... linkn` are the names of the links you want to add to the group. The links are already defined.

DISC



Purpose

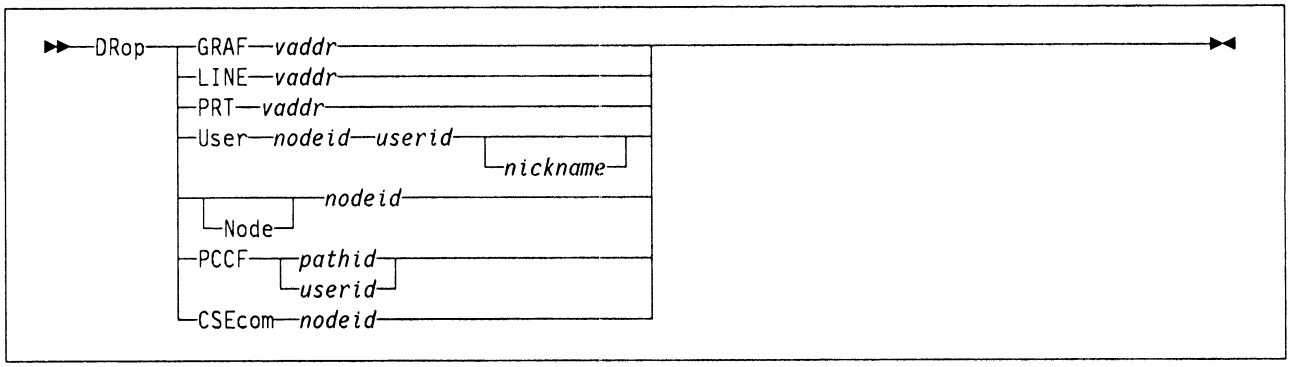
Use the DISC command to disconnect the PVM operator's display station. Use this command rather than the CP DISC command. Using the DISC command allows system spool files to be properly maintained.

Usage Notes

1. This command is restricted to a PVM operator logged onto the PVM user ID.
2. If the PVM AUDIT command or configuration file record had been previously specified, all "writes" or output messages, including those from PVM, sent to your virtual console will continue to be captured. If PVM tracing to a virtual printer has been started, records will continue to be added as appropriate.
3. If you log onto PVM, disconnect while SET RUN is OFF, then reconnect, you will not see any log messages or a RECONNECTED message. You will only see the CP READ status. The messages will appear after you issue CP BEGIN.

   However, if you disconnect while SET RUN is ON, then reconnect, you will receive the log messages and a RECONNECTED message.

DROP



Purpose

Use the DROP command to terminate:

- An attached display support task
- A line
- An attached printer support task
- A user session
- A specific PCCF path
- A PCCF support task started by a START PCCF command
- A CSE communication task.

Operands

GRAF *vaddr*
immediately terminates the specified attached display support task and any active user session.

The *vaddr* is the virtual address of the attached display support task terminated. The address specified must be the same as the address used to start the corresponding attached display support task.

LINE *vaddr*
immediately terminates the specified line and any active user sessions associated with the link.

The *vaddr* is the virtual address of the line terminated. The address specified must be the same as the address used to start the corresponding line.

PRT *vaddr*
immediately terminates the attached printer support task for the specified printer. The printer session is terminated and the attached printer support task is deactivated.

The *vaddr* is the virtual address of the 3270 Information Display System printer supported by the dropped task. The address specified must be the same as the address used to start the corresponding attached printer support task.

User *nodeid userid nickname*
terminates the specified user session. The user is notified of impending system action. The *nodeid* is the identification of the user's originating node

where the user's session is terminated. The *userid* is the identification of the user whose PVM session is terminated.

The session identification is:

- GRAF*xxx* for PVM dialed users
- the user's user ID for PASSTHRU users (or if you wish to drop all of an MPVM user's sessions)
- *SES*nnnn* for an MPVM session
- CL*cc*-P*pp* for a remote cluster printer or user attached to a PVM node
- **nnnnnnn* for a PCCF path or user
- Real address of a display station that is locally attached to a PVM node,
- an LU name, if the session origin is an SNA application.

For MPVM users, if the *nickname* is supplied, the MPVM session will be dropped if the specified *userid* and *nodeid* are the user's origin VM ID and node ID. A user's session can also be dropped by supplying the PVM user ID for the session (in the form of *SESnnnn) in the *userid* parameter. In this case, the nickname can be omitted. All of a user's sessions can be dropped by supplying the user's origin VM ID and node ID and omitting the nickname option.

Node *nodeid*
immediately terminates the specified PVM link. Any active user sessions associated with this link are also terminated. The *nodeid* is the node name defined in the PVM configuration file.

PCCF *pathid*
terminates the PCCF path identified by the PCCF path number *pathid*. If the path is active, it is broken immediately by simulating a SEVER external interrupt to the associated PCCF task.

PCCF *userid*
terminates a PCCF support task. The *userid* is the virtual machine for which PVM is handling PCCF connections. If there are any active connections, the task is not terminated until all connections are terminated.

CSEcom *nodeid*
ends a CSE communications task. The *nodeid* is the node name of the system for which the task is to be ended.

Usage Notes

1. This command is restricted to the PVM operator or an alternate operator specified on an AUTHORIZ record in the PVM configuration file.
2. If there are no sessions left for the MPVM user at the completion of the drop user command, the MPVM user's task will be removed from the PVM virtual machine.
3. To drop a remotely attached printer session, issue the following command:

   DROP User *nodeid userid*

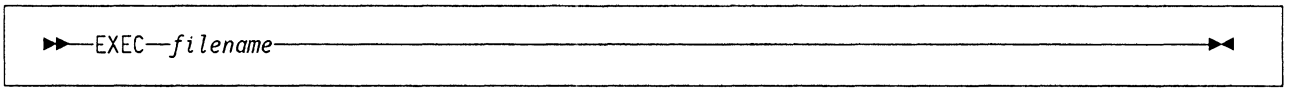
   nodeid
   is the link type of the R3270 *nodeid*.

userid
is CL*cc*-P*pp*, where *cc* is the cluster address specified on the CLUSTER record, and *pp* is the port address specified on the CLPORT record.

Note: You can use the DROP User command to attempt to restart a remote printer session. Issuing the DROP User command will cause the printer session to end, and then try to restart a new session.

4. When issued from another virtual machine using SMSG, you can use a command prefix with this command to tell PVM how to respond to the command. See "Using Command Prefixes" on page 255 for more information about command prefixes.

EXEC



Purpose

Use the EXEC command to execute all PVM commands in the CMS file specified on the command line. The file type of this file must be PVM.

Operands

filename
is the name of the file (file type PVM) to be executed.

Usage Notes

1. This command is restricted to the PVM operator or an alternate operator specified on an AUTHORIZ record in the PVM configuration file.
2. The file may contain any valid PVM commands. The file is processed by reading and stacking all the records within the file and then processing the records as individual commands.

   The file can have a fixed or variable record format with a maximum logical record length of 140 bytes. CP commands (for example, CP MSG) are truncated to 240 characters (to conform to the maximum length for a CP DIAGNOSE code X'08'). If the record format is fixed and the logical record length is from 72 to 80, PVM executes only the first 72 bytes of each record as the command. Columns 73 through 80 are ignored. This allows for serialization of the file.
3. When issued from another virtual machine using SMSG, you can use a command prefix with this command to tell PVM how to respond to the command. See "Using Command Prefixes" on page 255 for more information about command prefixes.

LMSG

Purpose

Use the LMSG command to send a message to all PVM users associated with the local PVM, or a specific link or group. If the name is the local PVM node name, the message is sent to all users who issued the DIAL, PASSTHRU, or MPVM command on the local system and selected the local node name. If the name is a link name, the message is sent to all users whose session is associated with the specified link. If the name is a group name, the message is sent to all users whose sessions are associated with any link in the group.

When the message arrives at the specified user's system, it replaces the current image on the user's display. The user recalls the stored image by pressing any interrupt key. The sender's node ID and VM ID are displayed with the message at the receiving location(s).

Operands

linkid
is the local PVM node name, a link defined on the local PVM system, or a group defined on the local PVM system.

message
is the text of the message sent. The message can be of any format and characters acceptable to the CP command processor. The total length is limited to the CP command input limit.

Usage Notes

1. This command is restricted to the PVM operator or an alternate operator specified on an AUTHORIZ record in the PVM configuration file.
2. No error message is issued if the specified link has no sessions.
3. When issued from another virtual machine using SMSG, you can use a command prefix with this command to tell PVM how to respond to the command. See "Using Command Prefixes" on page 255 for more information about command prefixes.

LOGMSG

Purpose

The operator uses the LOGMSG command to enter the variable data text that appears in the LOGMSG area on both the PVM Selection Menu and the Initial Menu.

Any entry of the LOGMSG command overlays the previous LOGMSG text. Issuing a LOGMSG command with no text overlays the previous LOGMSG text with blanks.

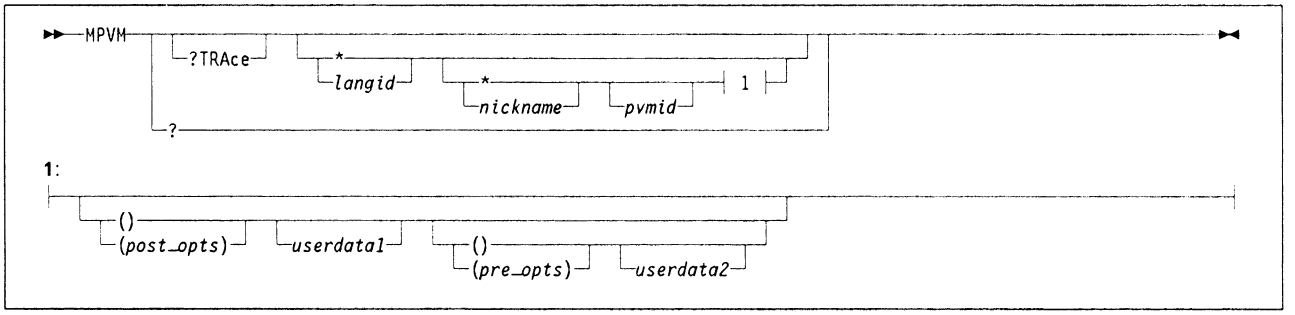
Operands

message
is the text of the message. You may use any characters. This field may be from 0 to 125 characters long, using two lines. If you use the shortest form of the LOGMSG command, LOGM, the message could be up to 125 characters long. If you use the longer form, LOGMSG, the message could only be up to 123 characters long.

Usage Notes

1. This command is restricted to the PVM operator or an alternate operator specified on an AUTHORIZ record in the PVM configuration file.
2. When issued from another virtual machine through SMSG, you can use a command prefix with this command to tell PVM how to respond to the command. See "Using Command Prefixes" on page 255 for more information about command prefixes.

MPVM



Purpose

Use the MPVM command to invoke MPVM. If used without any operands, a panel is displayed showing all session nicknames defined in the MPVMSYS NAMES and MPVM NAMES files.

Note: The MPVM command is invoked when MPVM is issued from CMS for the first time. This command calls the MPVM macro using the parameters exactly as given on the original MPVM command line. Subsequent use of MPVM from within the established MPVM environment (such as `?MPVM LOCAL2`) calls the MPVM macro directly for establishing additional sessions.

Operands

? invokes the VM/ESA HELP Facility for the MPVM command.

?TRAce
Sets the MPVM TRACE on immediately, before session initialization is started.

langid
use the *langid* parameter to specify the language for MPVM to use for this session. The *langid* is a 1-to-5-character identifier. If the default language is desired and the user wishes to select a session from the command line instead of the selection menu, an asterisk (*) must be used as a place holder. For example, to invoke the MPVM session `PAYROLL`, type:

```
MPVM * PAYROLL
```

nickname
is the 1-to-8-character nickname of the session to establish if the session is not already established. If the session is already active, the name is selected. If you want to specify the next parameter, but not a nickname, you must use an * here.

The value specified by the *nickname*:

- Cannot start with an escape (default value ?) character
- Must contain only the characters a—z, A—Z, 0—9, and the following special characters:
  . , _ : % & @ # ?
- Must be unique to all currently established sessions for this user
- Cannot be the word *CANCEL*.

pvmid
is the user ID of the PVM server virtual machine to which you wish to connect. Your system could include more than one PVM server virtual machine; however, once you establish an initial connection with a PVM server virtual machine, all sessions must be through that same PVM server machine.

post_opts
specifies MPVM variable name/value pairs that are processed *after* tags in MPVM's NAMES file(s). This allows you to override MPVM variables settings that were defined in the NAMES file(s). These *post_opts* must be specified in the format:

(*variable_name value*)

userdata1
specifies additional user information that the MPVM command does not parse. You can use this to *pass* data to another MPVM macro; that macro can use the *EXTRACT /ORIGCMD* subcommand to get the entire command string that was supplied on the MPVM command. *userdata1* and *userdata2* will be part of that command string.

To specify *userdata1*, you must code parentheses (as placeholders) in the *post_opts* position, even if you do not specify any *post_opts*.

pre_opts
specifies MPVM variable name/value pairs that are processed *before* tags in MPVM's NAMES file(s). This allows you to define initial default settings for MPVM variables. These *pre_opts* must be specified in the format:

(*variable_name value*)

userdata2
specifies additional user information that the MPVM command does not parse. You can use this to *pass* data to another MPVM macro: that macro can use the *EXTRACT/ORIGCMD* subcommand to get the entire command string that was supplied on the MPVM command. *userdata1* and *userdata2* will be part of that command string.

To specify *userdata2*, you must code parentheses (as placeholders) in the *pre_opts* position, even if you do not specify any *pre_opts*.

Usage Notes

1. Note the following about the *post_opts* and *pre_opts* options:
   - MPVM variables that can only be extracted are not valid as *post_opts* or *pre_opts*.
   - You can use these options to give values to *most* MPVM variables that can be set. However, the following variables are exceptions because they have special values:
     - UPDATED
     - EXIT11IF through EXIT20IF.
   - You can set MPVM string variables in *post_opts* and *pre_opts*. If the string value is only one word, you do not need any delimiters. If the string value is more than a single word, or if a delimiter character is contained within the word, you must use a delimiter character before and after the string value. The delimiter character can't be used in the string itself, and it must be something other than 0-9, a-z , or A-Z.

You *can* use parentheses within string variables in *post_opts* and *pre_opts*.

- You can use the *post_opts* to override variable settings for the current session. To do this, you issue the MPVM macro using the nickname of the current session.
- When you use the MPVM command to reconnect after a temporary disconnect, *post_opts* variables that must be set before session initialization are ignored. This is so that you can use the same MPVM command syntax to reconnect that you used with your initial MPVM connection.
- Specifying the MPVM variables EXIT11IF through EXIT20IF as MPVM command line options can be complex, because you must use delimiters within delimiters. Consider the following example:

  ```
  mpvm * mysess (exit11if /"VM READ"3263/ exit11 /MSG There is a VM READ/
  ```

  As a MPVM command option, the *entire* EXIT11IF variable's string value must be surrounded by delimiters; in addition, the part of the EXIT11IF value that represents the string being sought (`VM READ`) must be delimited from the area specification (`3263`).

2. If you need to enter `mpvm` with only a left parenthesis and some postopts, enter

   ```
   MPVM * (TRACE ON
   ```

   instead of entering

   ```
   MPVM (TRACE1 ON
   ```

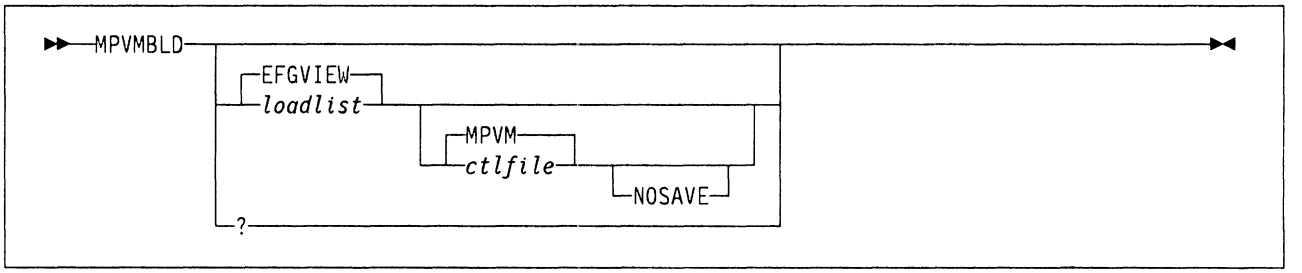
   This ensures that there is no confusion about the *langid* parameter placeholder.

Messages

Here are the possible return codes you can get from issuing the MPVM command:

-3	MPVM failed due to an installation error
0	Normal exit after all sessions are terminated by SESSTERM
1	Temporarily disconnected
3	You were idle too long (MAXIDLE setting exceeded)
4	User quit from an initialization prompt
5	Missing or invalid operand
8	This is an *error* return code, with a different meaning for each MPVM subcommand.
9	File I/O error
10	Terminal I/O error, or error in 3270 data stream
11	IUCV communications error with PVM server
12	Unable to establish connection to PVM server
15	The last (or only) session was terminated by PVM server without request
16	No current session was initially established or session termination occurred during INITCMD processing
18	The PVM server is in the process of shutting down, or in quiesce mode; no new connections are allowed
19	Maximum number of users are already connected to the PVM server
25	Insufficient free storage available
50	Illegal recursion into the MPVM user module
970	Unable to access MPVM common disk
998	IUCV protocol error
999	Internal logic error

MPVMBLD (PVM 2.1)



Purpose

Use the MPVMBLD command when regenerating the MPVM user module.

Operands

loadlist
specifies the name of an exec file that contains the names of object modules in the order in which they are to reside in the resulting module. The default value is **EFGVIEW**. The module created has the same file name as the loadlist.

The command supplied on the distribution tape may be modified to customize it for the installation.

ctlfile
specifies the file name of the control file (file type CNTRL). The default value is **MPVM**.

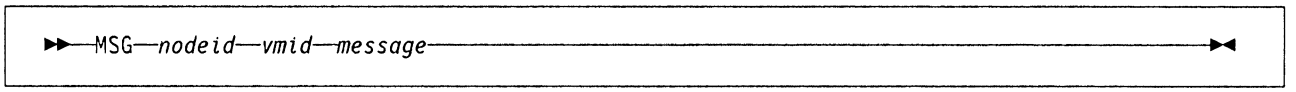
NOSAVE
specifies to replace the original files with the new files (file type MODULE, MAP, and PRELMAP). The default is to save the current files by prefixing the file names of the original files with an *O* and then saving them prior to creating new files.

? displays the valid parameter format at the user's display station.

Usage Notes

1. This command is intended for use by the PVM system programmer.

MSG



Purpose

MSG causes a line of text to be sent to the specified node and presented to a user or operator using the CP MESSAGE command.

Operands

nodeid
is the identifier of the target node that is to receive the specified message text line. This node can be the local node.

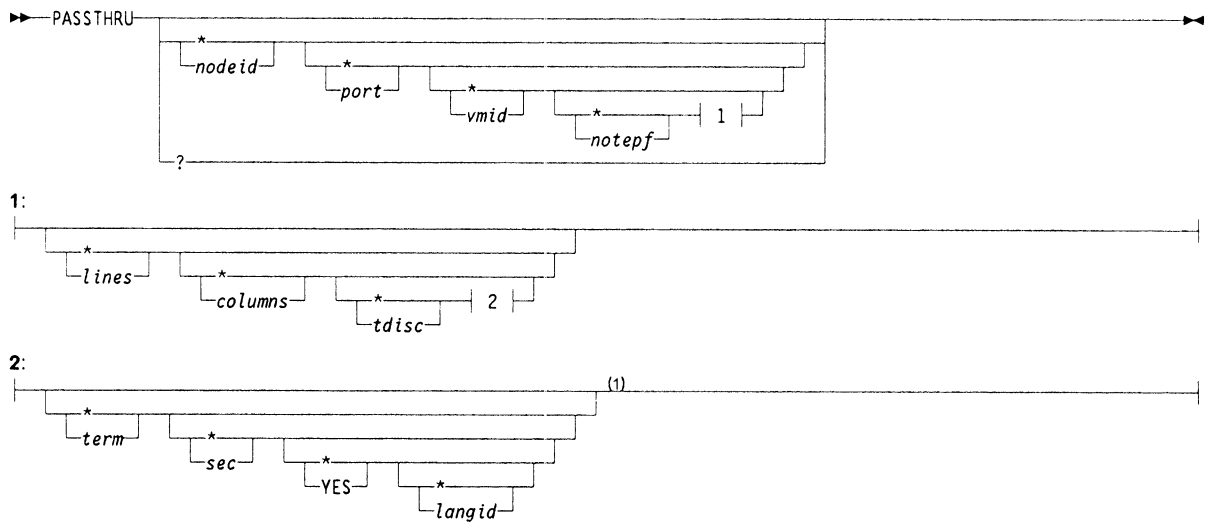
vmid
is the identifier of a logged on user at that location.

message
is the text of the message to be forwarded. The message can be of any format and characters acceptable to the CP command processor. The total length is limited to the CP command input limit.

Usage Notes

1. When issued from another virtual machine using SMSG, you can use a command prefix with this command to tell PVM how to respond to the command. See"Using Command Prefixes" on page 255 for more information about command prefixes.

PASSTHRU



Note:

[1] PASSTHRU is a positional command. If you wish to enter an option that follows other options that you do not wish to use, you must enter an asterisk (*) for each unused option that occurs before the option you wish to use. For example, if you wish to enter PASSTHRU with the *langid* operand, you must enter an asterisk as a placeholder for each preceding option. Your command would be PASSTHRU * * * * * * * * * *langid*. The ? operand cannot be followed by any other operand, since PASSTHRU ? provides help information about the PASSTHRU command.

Fragments *1* and *2* refer you to a continuation of the positional, sequential, operands. When you have completed entering operands, you can follow the syntax diagram to the end of the command, bypassing any remaining optional operands.

Purpose

You can only issue the PASSTHRU command from a 3270-type display station. If you are logged on to CMS, you can establish initial contact with PVM by issuing the PASSTHRU command. The target system you wish to interactively communicate with can be specified by including the target node identifier on the command line. If you do not specify a node, you can select a node from the PVM Selection Menu, which will be displayed. The PVM session is active until the user terminates the target application session and exits from PVM.

The PASSTHRU command can execute in CMS subset mode on systems with CMS release 5 or higher. You can also set defaults for the various parameters and use nicknames for nodes.

Operands

? invokes the VM/ESA HELP Facility for the PASSTHRU command.

nodeid
is the node identification of the target system or remote 4300 processor. If you specify an *, the PVM Selection Menu is displayed to assist you.

port
is a specific port number (if the specified node is an emulator node). If you specify a value for *port*, you must also specify a value for *nodeid*. The value of *port* ranges from 0 to 31. Specifying an * (if you specify a *nodeid*) assigns the first available port that matches the user's display station type.

vmid
is the PVM virtual machine ID. If you specify *, the default (PVM) is used.

notepf
is the number of the program function key causing the currently displayed screen to be written to a CMS file named PASSTHRU DATA on the A-disk. Acceptable values are 1 through 24. Specifying an * bypasses the Notepad facility. You cannot use the same PF key for the Notepad, temporary disconnect (*tdisc*), or session terminate (*term*) functions. The Notepad facility is available only if you specify a value for *notepf* on the PASSTHRU command line.

lines
is the number of lines from the currently displayed screen written to a CMS file by the Notepad facility, starting from the top of the screen down. Specifying an * results in a value of the screen size minus 2. Acceptable values are 1 through *nn*, *nn* being the maximum number of lines on the screen of the accessing device. If you specify a value for *lines*, then you must also specify a value for *notepf*.

columns
is the number of columns from the currently displayed screen written to a CMS file by the Notepad facility, starting with the leftmost column on the screen. Specifying an * results in a value of the screen width. Acceptable values are 1 through 132 for 3278 Model 5, or 1 through 80 for other display stations. If you specify a value for *columns*, then you must also specify a value for *notepf*. If a PASSTHRU DATA A1 file already exists, its record length must be equal to the value specified in *columns* or the file does not accept Notepad records. Otherwise, if the record length does not equal the value in *columns*, the file must be renamed or erased. A new PASSTHRU DATA A1 file, with the correct record length, is created the next time the Notepad facility is used.

tdisc
is a sequence entered to temporarily disconnect from a PVM session. The sequence can be a 1-to-8-character string (other than a single *), a PF function key, or a PA function key.

You may remain disconnected up to the disconnect time limit specified by the installation. Your screen image is saved and then restored after the PVM session is resumed. If you enter the disconnect sequence while the Selection Menu is displayed, the PVM session is terminated (temporary disconnect is not activated in this case).

If you specify *, the temporary disconnect function is not available for that session.

The valid range for PF keys is 1 to 24. Ensure that the device supports the specified PF key before assigning a function to it. PF, PF0, and PFnnn, if specified, are treated as a string. You cannot use the same character string, PF key, or PA key for the temporary disconnect, session termination, or notepad (notepad may only be assigned to a key, not to a string) functions.

term
is a sequence used to terminate a PVM session (following a disconnect from a target application). The sequence can be either a 1-to-8-character string, a PF function key, or a PA function key.

This mode of session termination is not required if the target node is a PVM virtual machine. In that case, the session automatically terminates when the user logs off or disconnects from the target application.

Specifying an * results in the PVM system default string. Each display station input is tested for the terminate sequence, and if found, the PVM session terminates.

The valid range for PF keys is 1 to 24. Ensure that the device supports the specified PF key before assigning a function to it. PF, PF0, and PFnnn, if specified, are treated as a string. You cannot use the same character string, PF key, or PA key for the session termination, temporary disconnect, or notepad (notepad may only be assigned to a key, not to a string) functions.

sec
is a PVM secondary routing node name. If specified, PVM routes the session through the specified node. This is particularly useful if the usual route to your target node is not available.

YES
activates I/O verification for this PVM session. Any other value indicates no verification. The default is no verification.

Note: Specifying **YES** for verification can cause a significant degradation of performance in your PVM session.

langid
identifies the language that PVM will use for the session. If a value for *langid* is not specified, the language used in the PVM session is determined as follows:

1. The language being used for the user's CMS session, except when PVM is not enabled for that language or when the language is a DBCS language and the user's display station cannot handle DBCS
2. The default PVM language except when that language is a DBCS language and the user's display station cannot handle DBCS
3. The PVM backup language.

Usage Notes

1. You can use CP and CMS facilities to define your own defaults for PASSTHRU. For example, you can use:
   - Your *userid NAMES* file to define nicknames for target systems
   - A CMS SYNONYM file to set synonyms for the PASSTHRU command
   - The CP SET PFn command to set a PF key to a PASSTHRU command
   - CMS GLOBALV to set up defaults for PASSTHRU parameters
   - An exec written to issue a regularly used or complex PASSTHRU command.

2. To find out what the default session terminate string on your PVM system is, enter:

   ```
   passthru
   ```

   to display the PVM Selection Menu. The default is displayed in the `End Session` field.

Examples

Defaults or PF keys can be set so that PASSTHRU can be issued without having to type in values for all the parameters. To make PASSTHRU easier to use, you can:

- Set up nicknames for target systems you can access by using your CMS *userid NAMES* file. Within the CMS NAMES file, you can use a *:NODE* or *:PVMNODE* tag which PASSTHRU uses but other programs ignore. For example, the name *HARTFORD* could be a nickname for the system *CONN1* and *NYC* could be a nickname for *NEWYORK*. Entries in your CMS NAMES file could look like the following:

  ```
  :nick.HARTFORD    :node.CONN1       ...
  :nick.NYC         :pvmnode.NEWYORK  ...
  ```

- Set a synonym for the PASSTHRU command in a CMS SYNONYM file. For example, if you want *PASS* to be a synonym for PASSTHRU, enter in your SYNONYM file:

  ```
  PASSTHRU  PASS
  ```

  For details on CMS SYNONYM, see the *VM/ESA CMS Command Reference*.

- Set a PF key to the command needed to start a PVM session. For example, to set PF01 to start a PVM session and as a temporary disconnect key, you could enter:

  ```
  cp set pf01 immed passthru newyork * * * * * pf01
  ```

  Pressing PF01 will immediately take you to the *NEWYORK* system. Pressing PF01 from within the *NEWYORK* system temporarily disconnects you from that system, and returns you to your host system. By adding this to your CMS PROFILE EXEC, PF01 will be set to the above setting every time CMS is IPL'd. For details on the CP SET command, see the *VM/ESA CP Command and Utility Reference*.

- Set up your own defaults for PASSTHRU parameters with the CMS GLOBALV command. For example, to initialize default values for Notepad (the fourth, fifth, and sixth parameters on the PASSTHRU command) and a Temporary Disconnect string (the seventh parameter), enter:

  ```
  globalv select passthru setlp passthru * * * 12 24 80 pf01
  ```

  Thereafter, your personal defaults apply when you enter the PASSTHRU command. You could enter

  ```
  passthru newyork
  ```

  to connect to NEWYORK with your personal defaults in effect for Notepad and Temporary Disconnect.

  You can also override your personal defaults. Just use two asterisks (**) in place of those parameters that you want to keep as your personal default. For example, to start a PVM session with PF3 set to Temporary Disconnect, enter:

  ```
  passthru newyork ** ** ** ** ** pf3
  ```

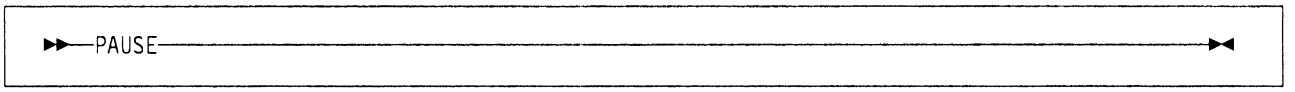
You can also use the GLOBALV command to see your personal defaults for the PASSTHRU command. To display the default values for the above PASSTHRU example, you would enter:

```
globalv select passthru list passthru
```

See the *VM/ESA CMS Command Reference* for details on the CMS GLOBALV command.

- Set your own defaults for PASSTHRU parameters within a CMS EXEC. See the *VM/Pass-Through Facility User's Guide* for details.

PAUSE



Purpose

Use the PAUSE command to cause the command processor to enter the wait state for one second. You would normally use the PAUSE command in a PVM exec. This allows any previous commands to complete execution.

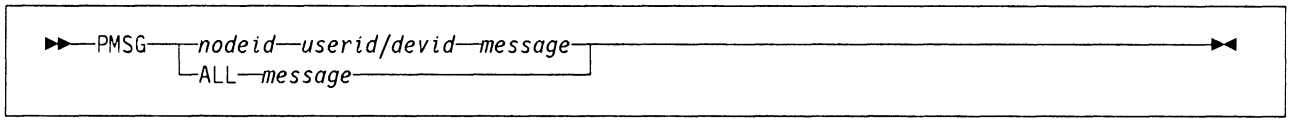
Many of the PVM commands are only partially processed by the command processor. The START command, for instance, is pre-checked by the command processor and then posted to the session manager to complete processing. Because the command processor runs at a higher priority then most other tasks, these type commands do not complete until the command processor has no more work to do.

For example, if you placed in a PVM exec a DROP command followed by a START command, the START command would not be executed because the DROP command had not completed processing. The PAUSE command allows the sequence to work properly by delaying the command processor to give time for the DROP to complete. The DROP command to some line drivers can take longer than the PAUSE time interval. In this case, more than one PAUSE command can be added.

Usage Notes

1. This command is restricted to the PVM operator or an alternate operator specified on an AUTHORIZ record in the PVM configuration file.
2. When issued from another virtual machine using SMSG, you can use a command prefix with this command to tell PVM how to respond to the command. See "Using Command Prefixes" on page 255 for more information about command prefixes.

PMSG



Purpose

Use the PMSG command to send a message to a particular user or all users who have sessions with or through a PVM node. When the message arrives at the specified user's system, it replaces the current image on the user's display. The user recalls the stored image by pressing any interrupt key. The sender's node ID and VM ID are displayed with the message at the receiving locations.

Operands

nodeid
is the identification of the target user's node when the message is forwarded to a single user.

userid/devid
is the identification of the user (*userid*) or the identification of the user's attached display support task (*devid*) when the message is forwarded to a single user. This command is not valid to printer devices.

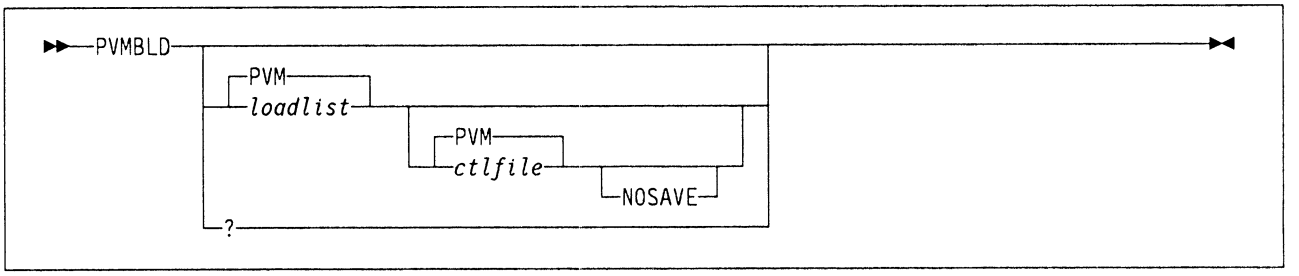
ALL
forwards the message to all PVM users whose sessions originate, terminate, or pass through this PVM.

message
is the text of the message forwarded. The message can be of any format and characters acceptable to the CP command processor. The total length is limited to the CP command input limit.

Usage Notes

1. The **ALL** option of this command is restricted to the PVM console operator or an alternate operator specified on an AUTHORIZ record in the PVM configuration file.
2. When issued from another virtual machine using SMSG, you can use a command prefix with this command to tell PVM how to respond to the command. See"Using Command Prefixes" on page 255 for more information about command prefixes.

PVMBLD



Purpose

Use the PVMBLD command when regenerating the PVM module.

Operands

loadlist
specifies the name of an exec file that contains the names of object modules in the order in which they are to reside in the resulting module. The default value is **PVM**. The module created has the same file name as the loadlist.

The command supplied on the distribution tape may be modified to customize it for the installation.

ctlfile
specifies the file name of the control file (file type CNTRL). The default value is **PVM**.

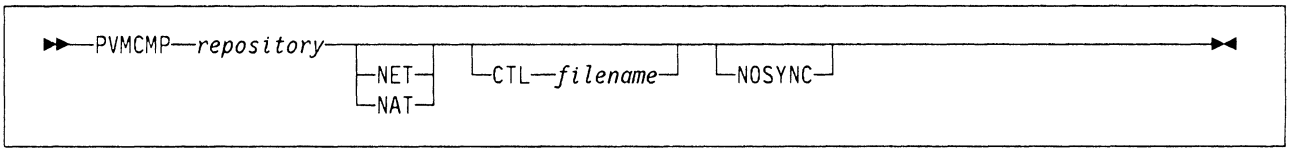
NOSAVE
specifies to replace the original files with the new files (filetype MODULE, MAP, and PRELMAP). The default is to save the current files by prefixing the file names of the original files with an *O* and saving them prior to creating new files.

? displays the valid parameter format at the user's display station.

Usage Notes

1. This command is intended for use by the PVM system programmer.

PVMCMP



Purpose

Use the PVMCMP command to compile a PVM message repository.

Operands

repository
The file name of the repository that is to be compiled. DVM*langid* is used for PVM and PVMG messages. EFG*langid* is used for DVMUSI and MPVM messages.

NET
Indicates that the repository is to be compiled for use as the backup language. NET is the default when the repository is for the UCENG language.

NAT
Indicates that the repository is to be compiled for use as a national language. NAT is the default when the repository is for a language other than UCENG.

CTL *filename*
Indicates that the repository is to be updated before it is compiled using the CMS UPDATE command and the control file that is identified by *filename*. (The control file must have a file type of CNTRL.)

NOSYNC
Is used primarily by IBM Translation Centers. NOSYNC forces message text expansion rules for English repositories to allow for translation. $LISTING and $COMP*nnn* files are produced, but the PVMDEST FILE is not read and object and $SYNC*nnn* files are not produced.

Usage Notes

1. This command is intended for use by the PVM system programmer.
2. Files created by this command include:
   - A repository object code file.

     If the CTL option is not used, this file is named DVM*langid* TEXT or EFG*langid* TEXT. If the CTL option is used, then PVMCMP uses the update level identifier in the CNTRL file to name the file type of the output file. If the update level identifier of the last located update file is anything other than TEXT, the update level identifier is prefixed with the characters TXT to form the file type. If no update files are found the file type is named TEXT.
   - A listing file named DVM*langid* $LISTING or EFG*langid* $LISTING.

     This file contains an input definitions plus any syntax error messages.

- A comparison file named DVM*langid* $COMP*nnn* or EFG*langid* $COMP*nnn*, where *nnn* is the highest compatibility level for which items were defined in the repository with a PVM release.

  This file contains all of the nontranslatable information in the repository, not counting information added with a SERVICE control line. By comparing it to the corresponding, upper case English comparison file (supplied by IBM), it verifies that none of the nontranslatable information in the repository has been changed.

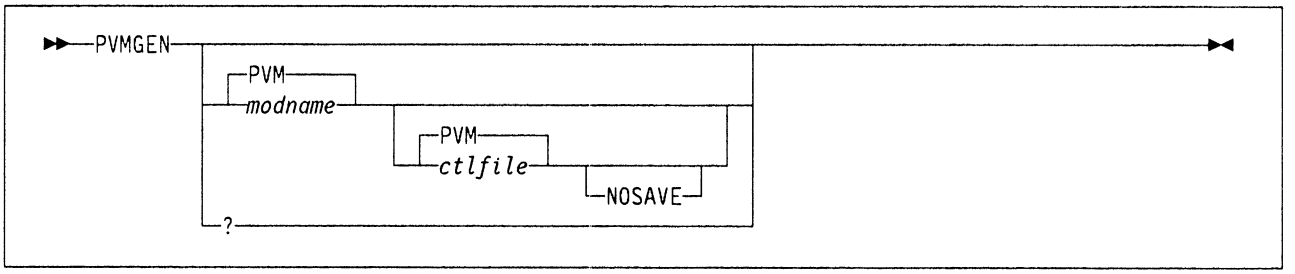
- When compiling an uppercase English repository file (UCENG), the comparison file for that repository must not be accessed in file mode A.

- Except when the NOSYNC option is specified, PVMCMP produces a file named DVM*langid* $SYNC*nnn* or EFG*langid* $SYNC*nnn*, where *nnn* is the highest compatibility level encountered in the repository.

  This file verifies that the object files are synchronized with respect to their common messages. The repositories are synchronized if DVM*langid* $SYNC*nnn* and EFG*langid* $SYNC*nnn* are identical.

3. If the CTL option is specified, PVMCMP uses the CMS UPDATE command to process any update files for the message repository and PVMDEST FILE. PVMCMP does not erase any files created by CMS UPDATE.

PVMGEN (PVM 2.1)



Purpose

Use the PVMGEN command for generating the DVMUSI module that executes in the user's CMS virtual machine when the PASSTHRU command is issued.

Operands

modname
specifies the name of the module to be created. The default value is DVMUSI. There must be a loadlist exec with the same name specified for *modname* plus an *E* appended to it. The loadlist exec contains the names of the object modules in the order in which they are to reside in the resulting module.

ctlfile
specifies the control file name (file type CNTRL). The default file name is PVM.

NOSAVE
specifies to replace the original files with the new files (file type MODULE, MAP, and PRELMAP). The default is to save the current files by prefixing the

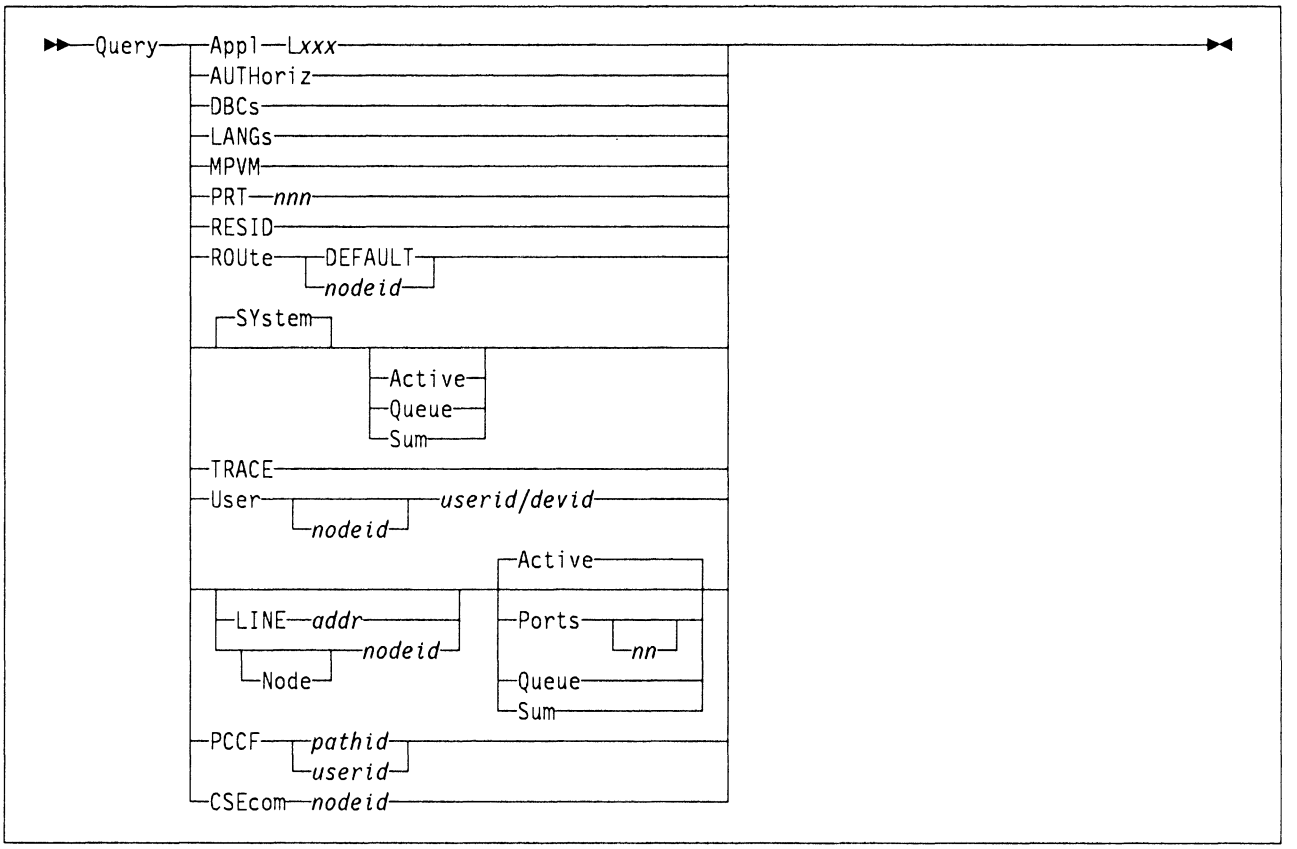
file names of the original files with an *O* and then saving them prior to creating new files.

? displays the valid parameter format at the user's display station.

Usage Notes

1. This command is intended for use by the PVM system programmer.

QUERY



Purpose

Use the QUERY command to display one of the following:

- Users authorized to issue restricted PVM commands
- System status
- Current status of each line supported by PVM
- Current status of a logical device
- Current status of ports on a line
- Contents of the routing table
- Current value accumulated in the error counter for a specified line
- Current session status of a particular printer supported by PVM
- User session information
- Current DBCS settings
- National languages enabled on a PVM system
- A specific PCCF path
- All PCCF paths for a specific virtual machine

- Current settings and statistics for MPVM users
- Current status of a CSE communications task
- APPC resource identifier defined in the PVM configuration file
- Current TRACE settings.

If you enter the QUERY command with no operands, QUERY SYSTEM is the default.

Operands

Appl L*xxx*
displays the status of the session associated with the local logical device L*xxx*. The value for *xxx* is a hexadecimal number between X'000' and X'FFF'.

AUTHoriz
displays the current list of authorized users in the system who can enter PVM restricted commands.

DBCs
displays the DBCS settings for CP operator, the PVM console (and the AUDIT file), and other VM users when the default language for the PVM system is a DBCS language.

LANGs
displays the language identifiers of the languages for which PVM is enabled. LANGS also displays the languages used as the default national language and as the backup language at the local node.

MPVM
displays the current number of MPVM users, the highest number of MPVM users, the current total number of MPVM sessions, and the highest total number of MPVM sessions. Also displayed are the current settings for maximum idle, maximum suspend and temporary disconnect times, maximum number of sessions per user, and maximum number of MPVM users allowed in the PVM server.

PRT *nnn*
displays the status of the printer associated with the virtual address specified by *nnn*. One of the following is displayed:

ACTIVE	Initiated, but connection not completed.
CONNECT	Established, connection complete.
not ready	Pending manual intervention to *ready* device.

RESID
displays the unique resource identifier defined in the PVM configuration file.

ROUte
displays the contents of the local PVM's routing table for either the DEFAULT route or the particular route name.

DEFAULT
indicates if there is DEFAULT routing. If DEFAULT routing exists, the node being routed to is displayed.

nodeid
The node's entry from the routing table is displayed.

SYstem
displays information about the status of the system. If you do not specify any other operands, the status of each link defined within the system is displayed. Query SYstem is the default value if you enter the QUERY command with no operands.

Active
displays the status of each active link only. If there are no active links, only the name of the local system is displayed.

Queue
displays the status and the associated user sessions for each active link.

Sum
displays the statistics of each line driver. This includes both transaction counts and any line error statistics.

TRACE
displays trace activities of PVM.

User *userid/devid*
displays information about the associated local user (*userid*) or the attached display (*devid*). The specified user must be connected to the local PVM server. The response includes the user's real display station address, type, and model number, and the session node and ID. *SES*nnnn* can be used to get information on a particular MPVM session. The number of sessions, with information on each session, will be displayed when querying an MPVM user.

User *nodeid userid/devid*
displays information about the associated session identified by *nodeid* and *userid/devid*. The specified session must be known to the local PVM. The response includes the user's real display station address, type, and model number, along with the session origin and destination IDs.

LINE *addr*
displays the status of the specified link.

Node *nodeid*
displays the status of the specified node.

Active
displays the status of the specified line, if active. Active is the default if Port, Queue, or Sum is not specified.

Ports *nn*
displays the status of port *nn*. If the port number is missing, all ports are displayed. If the link is a R3270 link, the associated ports for each cluster are displayed. This option is valid only for an emulator or R3270 link.

Queue
displays the status of the specified node and the associated user sessions for the specified node.

Sum
displays the statistics of the line driver. This includes both transaction counts and any line error statistics.

PCCF *pathid*
displays the PCCF path identified by the PCCF path number specified by *pathid*.

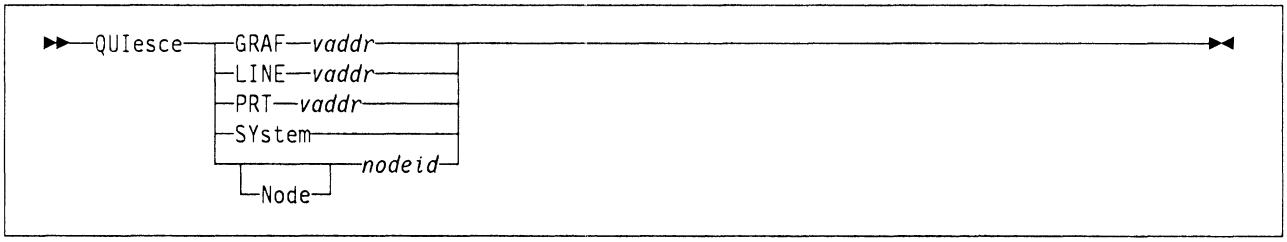
PCCF *userid*
displays all PCCF paths associated with the virtual machine identified by *userid*.

CSEcom *nodeid*
displays information about a CSE communications task. The *nodeid* is the node name of the system for which the status information is to be displayed.

Usage Notes

1. The AUTHorize operand is restricted and must be issued from a user ID on the local system.
2. When issued from another virtual machine using SMSG, you can use a command prefix with this command to tell PVM how to respond to the command. See"Using Command Prefixes" on page 255 for more information about command prefixes.

QUIESCE



Purpose

The PVM operator uses the QUIESCE command to quiesce:

- An attached display/printer support task
- A line driver task (such as the R3270 line driver)
- The PVM system.

Quiesce is a term used to describe the state of a task (or PVM itself). A task in a quiesce state does not allow any new sessions and terminates when all active sessions have ended.

The effect of the QUIESCE command can be reversed by issuing the RESUME command. Quiesce does not terminate any line drivers or attached display support tasks that have active sessions.

Operands

GRAF *vaddr*
indicates that the attached display support task *vaddr* is quiesced. A user currently dialed onto PVM is allowed to complete the session. At session termination the user is dropped, and the specified attached display support task terminates. If no user session is active, the specified task terminates immediately.

The *vaddr* specifies the virtual address of the attached display support task. The address specified must be the same as the address used to start the corresponding attached display support task (START GRAF nnn).

LINE *vaddr*
indicates that the line *vaddr* is quiesced. After issuing this command, no new users are allowed to initiate sessions through the associated *vaddr*. When the last user session on the line terminates, the line driver terminates. If no user session is active, the specified line terminates immediately.

The *vaddr* specifies the address of the line. The address specified must be the same as the address used to start the corresponding line.

PRT *vaddr*
indicates that the attached printer support task *vaddr* is quiesced. If the printer session was started but the connection never established, the session is terminated. If an established session has had no I/O activity for a 2-minute interval, the session is terminated.

The *vaddr* specifies the address of the attached printer support task. The address specified must be the same as the address used to start the corresponding attached printer support task (START PRT nnn).

SYstem
places the entire PVM system in a quiesced state. No new sessions can be initiated within or through this PVM system after issuing SYSTEM. This command does not terminate any line drivers, attached display support tasks, or user sessions.

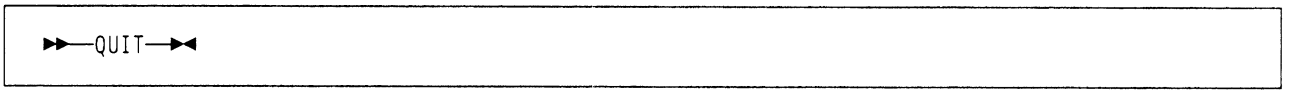
Node *nodeid*
indicates that link *nodeid* is quiesced. After you issure the command, no new users are allowed to initiate sessions through the associated *nodeid*. When the last user session on the line terminates, the line driver terminates. If no user session is active, the specified *nodeid* terminates immediately.

The *nodeid* must be the same name defined in the configuration file for the corresponding node.

Usage Notes

1. This command is restricted to the PVM operator or an alternate operator specified on an AUTHORIZ record in the PVM configuration file.
2. When issued from another virtual machine using SMSG, you can use a command prefix with this command to tell PVM how to respond to the command. See "Using Command Prefixes" on page 255 for more information about command prefixes.

QUIT



Purpose

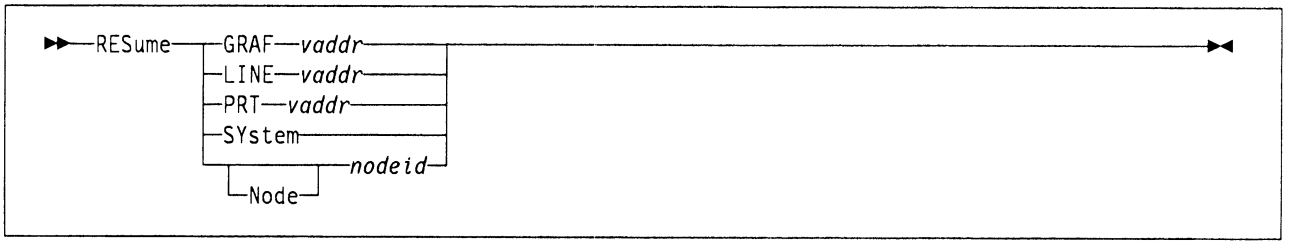
Use the QUIT command to terminate the PVM system immediately. No users are notified that the system is terminating. Not all allocated storage is freed when control returns to CMS. So CMS should be reinitialized, or an HX command should be issued to free allocated storage.

Before issuing the QUIT command, issue a DROP command for each active line or attached display support task so that all sessions are terminated.

Usage Notes

1. This command is restricted to a PVM operator logged on to the PVM user ID.

RESUME



Purpose

Use the RESUME command to reverse the effect of a previously issued QUIESCE command. As a result, the specified line, attached display/printer support task, or the entire system is made available.

Note: If the quiesced line driver or attached display/printer support task has already terminated due to the QUIESCE command, use the START command instead of the RESUME command.

Operands

GRAF *vaddr*
indicates that the attached display support task *vaddr* is not terminated when the current user's session ends. The *vaddr* specifies the address of the affected attached display support task. The address specified must be the same as the address used to start the corresponding attached display support task (START GRAF *vaddr*).

LINE *vaddr*
indicates that use of line *vaddr* is resumed. Following the issuing of this command, new users are allowed on the affected line. The previously issued QUIESCE command does not terminate the line. The *vaddr* specifies the address of the affected line driver. The address specified must be the same as the address used to start the corresponding line.

PRT *vaddr*
indicates that the attached printer support task *vaddr* is not terminated after all. The *vaddr* specifies the address of the affected attached printer support task. The address specified must be the same as the address used to start the corresponding attached printer support task (START PRT *vaddr*).

SYstem
indicates that the entire PVM system is to be resumed. New sessions may be initiated after this command is issued.

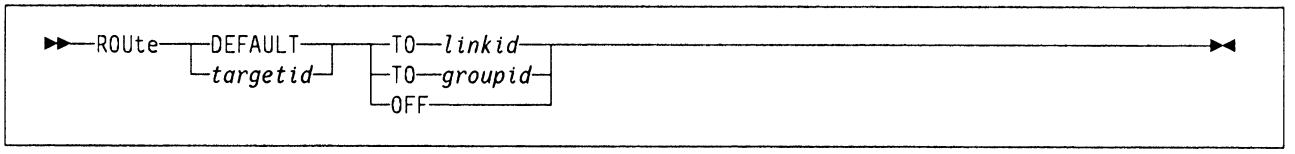
Node *nodeid*
indicates that the link whose node name is *nodeid* is resumed. After the command is issued, new users are allowed to initiate sessions on the affected line.

The *nodeid* must be the name defined in the configuration file for the corresponding line.

Usage Notes

1. This command is restricted to the PVM operator or an alternate operator specified on an AUTHORIZ record in the PVM configuration file.
2. When issued from another virtual machine using SMSG, you can use a command prefix with this command to tell PVM how to respond to the command. See "Using Command Prefixes" on page 255 for more information about command prefixes.

ROUTE



Purpose

The ROUTE command modifies the in-storage routing table used by PVM for session routing. Node names may be added, deleted, or their routing altered by the command.

Operands

DEFAULT
changes the DEFAULT routing.

targetid
is the target node added, deleted, or modified.

TO *linkid*
indicates that the target node is routed through the link *linkid*. *linkid* is the node name of the link. This node must be a PVM node directly connected to the local PVM system.

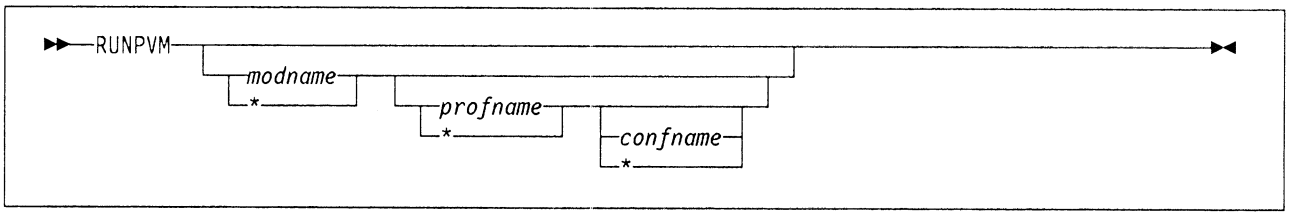
TO *groupid*
indicates that the target node is routed through the link group *groupid*. The *groupid* is the name of the link group. This group must be defined for the local PVM system.

OFF
removes the specified target node from the routing table. OFF may not be specified as a default route.

Usage Notes

1. This command is restricted to the PVM operator or an alternate operator specified on an AUTHORIZ record in the PVM configuration file.
2. When issued from another virtual machine using SMSG, you can use a command prefix with this command to tell PVM how to respond to the command. See "Using Command Prefixes" on page 255 for more information about command prefixes.

RUNPVM



Purpose

Use the RUNPVM command to load and start PVM.

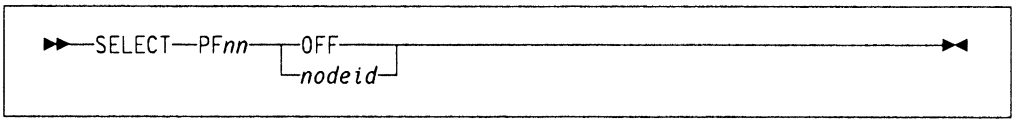
Operands

modname
is the name of the PVM module loaded. If you specify *, the default file name is PVM. The file type is MODULE.

profname
is the name of the PVM exec executed at startup time. If you specify *, the default file name, *profname*, is PVM. The file type of this exec is EXEC.

confname
is the name of the configuration file. If you specify *, the default file name, *confname*, is PVM. The file type of this configuration file is CONFIG.

SELECT



Purpose

Use the SELECT command to dynamically add, delete, or modify the nodes assigned to PF keys by the SELECT records of the configuration file.

Operands

PF*nn*
is the PF key modified. The value of *nn* ranges from 1 to 12, excluding 8.

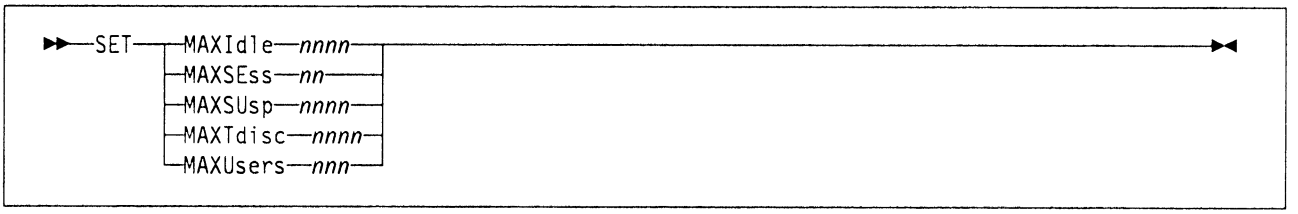
OFF
indicates that the PF key definition is to be reset.

nodeid
is the name of the node selected by the specified PF key. The node name should not be a member of a group. It can identify a group, but cannot identify an individual member of a group.

Usage Notes

1. This command is restricted to the PVM operator or an alternate operator specified on an AUTHORIZ record in the PVM configuration file.
2. When issued from another virtual machine using SMSG, you can use a command prefix with this command to tell PVM how to respond to the command. See "Using Command Prefixes" on page 255 for more information about command prefixes.

SET



Purpose

Use the SET command to change settings for MPVM users. These settings include the maximum:

- Time a MPVM user can remain idle
- Time a MPVM can have a session suspended
- Time a MPVM user can be temporarily disconnected
- Number of MPVM users allowed on the PVM server
- Number of sessions an MPVM user can have.

Operands

MAXIdle *nnnn*
The maximum number of seconds a MPVM user may remain idle before all of that user's sessions will be terminated by PVM. (Idle means there is no communication from the MPVM user to PVM for any session.) The *nnnn* ranges from 60 to 9998. A setting of 9999 will mean that there is no maximum idle time limit. PVM sends out a warning message when approximately three-fourths of the time interval has elapsed.

MAXSEss *nn*
The maximum number of sessions a MPVM user may establish. The *nn* ranges from 1 to 99. The maximum number of sessions allowed for all MPVM users combined is 4096.

Decreasing the maximum number of sessions allowed below the current number of sessions owned by a particular MPVM user is not going to affect those existing sessions; the user would not be able to start any more sessions.

MAXSUsp *nnnn*
The maximum number of seconds a MPVM user may have a session suspended before that user's session will be terminated by PVM. The value of *nnnn* ranges from 60 to 9998. A setting of 9999 will mean no maximum suspend time limit. PVM will send out a warning message when approximately one-half of the time limit has elapsed, and then again when three-fourths of the time limit has passed.

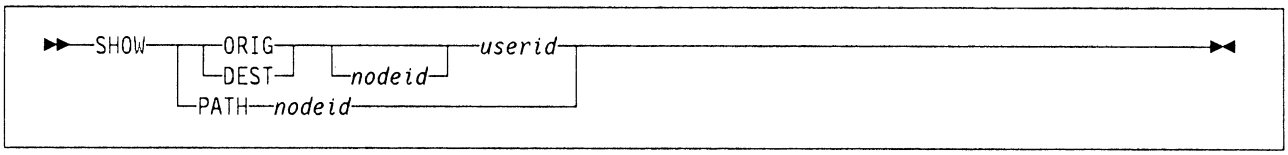
MAXTdisc *nnnn*
The maximum number of seconds a MPVM user may remain temporarily disconnected before all of the user's sessions will be terminated by PVM. The *nnnn* ranges from 60 to 9998. A setting of 9999 will mean no maximum temporary disconnect time limit. A setting of 0 will indicate that MPVM users may not temporarily disconnect. PVM will send out a warning message when approximately three-fourths of the time limit has elapsed.

MAXUsers *nnn*
The maximum number of MPVM users who may currently connect to PVM. The *nnn* ranges from 1 to 999. The maximum number of users can also be limited indirectly through the number of IUCV and APPC/VM paths supported by PVM. This value can be changed through the MAXCONN record in the configuration file but cannot exceed the limit specified in the CP OPTION directory record for the PVM virtual machine.

Usage Notes

1. This command is restricted to the PVM operator or an alternate operator specified on an AUTHORIZ record in the PVM configuration file.
2. When issued from another virtual machine using SMSG, you can use a command prefix with this command to tell PVM how to respond to the command. See "Using Command Prefixes" on page 255 for more information about command prefixes.

SHOW



Purpose

Use the SHOW command to show the path of a particular session, or the path to a particular node. The system on which the command is first executed locates the session and displays information indicating that it is part of the path. This message displays the link upon which the session comes in, and the link upon which the session goes out.

Operands

ORIG
traces the path to the origin. If you use ORIG and the origin is not the local system or the incoming link, the command is forwarded to the node connected to the incoming link for processing.

DEST
traces the path to the destination. If you use DEST and the destination is not the local system or the outgoing link, the command is forwarded to the node connected to the outgoing link for processing. In this way, the command is executed on each intermediate node along the path.

PATH
shows the PVM routing to get to a particular target node.

nodeid
is the node name of the origin or destination of the session. For ORIG or DEST, if you do not specify a value for *nodeid*; the local node name is assumed. For PATH, you must specify a value for *nodeid*, and it is the target node for which path information is required.

userid
is the PVM identification for the origin or destination of the session.

Usage Notes

1. SHOW PATH shows the PVM routing to get to a specific target node. The command finds the outbound path to the target node. If the target node is the local node or a non-network node, you receive messages that tell you the path to the target node, and then the message `COMMAND COMPLETE`. If the target node is not the local node or a non-network node, you receive the message `Path to node xxxxx is through node yyyyy`. The SHOW command is then forwarded to the node yyyyy for execution.
2. When issued from another virtual machine using SMSG, you can use a command prefix with this command to tell PVM how to respond to the command. See"Using Command Prefixes" on page 255 for more information about command prefixes.

SHUTDOWN

Purpose

Use the SHUTDOWN command to conditionally terminate the PVM server. If the PVM server has no active sessions, then PVM will shut down immediately. However, if the PVM server determines that there are active sessions, it will not shut down until all active sessions end. All line drivers and attached display support tasks are placed in a quiesced state. The attached display support tasks terminate as soon as the user session terminates. Line drivers terminate when the last users of the associated lines terminate their sessions. PVM terminates when the last user session terminates and all associated tasks terminate with it unless the IMMED option is specified. The operator can then wait for the sessions to terminate, or issue the SHUTDOWN IMMED command to shut down immediately.

During SHUTDOWN processing, an MPVM user task will be removed from the PVM virtual machine if there are no sessions for that user. If a MPVM user has any active sessions, these sessions are left active, but no new sessions are allowed. When the MPVM users end all sessions, their user tasks are removed from the PVM machine to allow PVM to shutdown.

Not all allocated storage is freed when control returns to CMS. Therefore, CMS should be reinitialized or an HX command should be issued to free allocated storage.

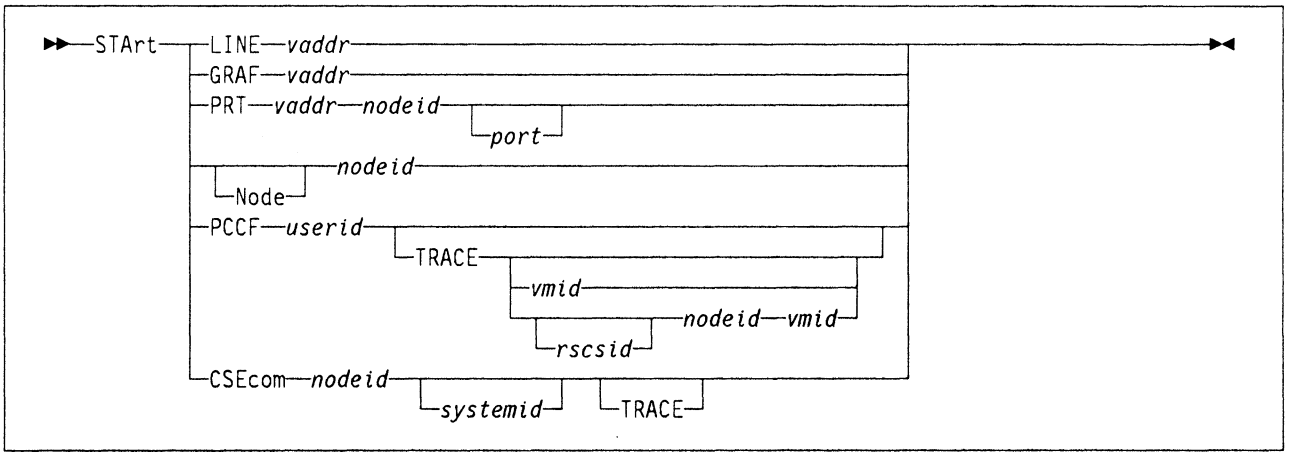
Operands

IMMED
indicates that the shutdown is immediate and all sessions are terminated.

Usage Notes

1. This command is restricted to a PVM operator, logged on with a PVM user ID.
2. For SHUTDOWN without the immediate option if:
   - A printer session was started but the connection was never established, the session is terminated
   - An established session has had no I/O activity for a two minute interval, the session is terminated.
3. For SHUTDOWN IMMED, sessions are terminated immediately.

START



Purpose

The PVM operator uses the START command (a restricted command) to start:

- A line driver task to a PVM virtual machine
- An IUCV line driver task to a PVMG virtual machine
- An attached display support task
- An attached printer support task
- A PCCF support task
- A CSE communication task.

Operands

LINE *vaddr*
starts a line driver at line *vaddr*. The address must be defined within the configuration file LINK record. If you are starting a line to a remote PVM connection using an ESCON channel, the address must be defined with a link type of 3088 within the configuration LINK record. The *vaddr* specifies the virtual address of the affected line driver.

GRAF *vaddr*
starts an attached display support task at virtual address *vaddr*. This address must have a value from X'20' to X'FFF'. The *vaddr* specifies the virtual address of the affected attached display support task.

PRT *vaddr nodeid port*
indicates that an attached printer support task is activated and a printer session is initiated with the printer at the specified virtual address. This address must have a value from X'20' to X'FFF'.

If the printer session cannot complete because a network path is down or is not connected, PVM attempts connection five more times at 30-second intervals and thereafter at 10-minute intervals. Retry continues until system shutdown, or until you enter DROP PRT or QUIESCE PRT.

vaddr specifies the virtual address of the affected attached printer support task.

nodeid indicates the target system node where the session is initiated. The target must be a non-PVM system accessed by 3271/3274 emulation.

port specifies a logical port assigned to the printer session. The specified port is assigned if not already in use and not reserved for another printer. The requested logical port must be one designated as a 3270 printer port by the PORT record in the configuration file. If a logical port is not specified, then the logical port assigned is either the port defined in the system configuration file for the printer's real address or the first available logical 3270 printer port.

Node *nodeid*
starts a line driver to the line whose node name is *nodeid*. The *nodeid* must be defined in the configuration file.

PCCF *userid*
starts a PCCF support task. The *userid* is the virtual machine for which PVM is to handle PCCF connections.

TRACE
Indicates that tracing for the PCCF support task is to begin immediately.

vmid
issues a CP SPOOL command to transfer output to the reader (*vmid*).

rscsid nodeid vmid
issues a CP SPOOL command and a CP TAG command to send the output over the RSCS network (*rscsid*) to the reader (*vmid*) at the specified node (*nodeid*).

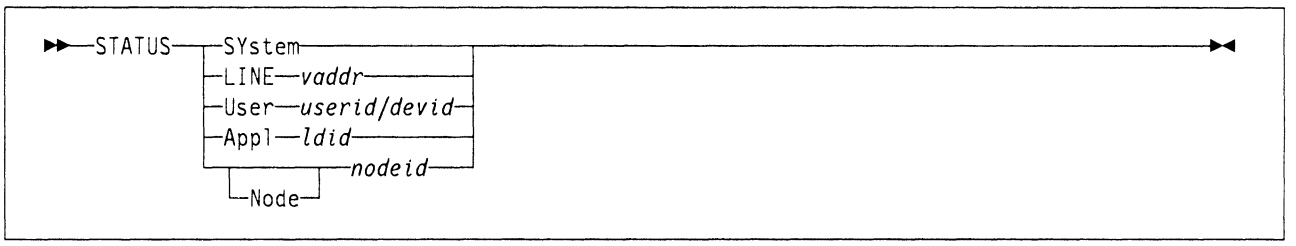
CSEcom *nodeid sytemid*
starts a CSE communications task. The *nodeid* is the node name of the system for which the session is to be started. The *nodeid* must be another CSE system. The *systemid* is the system name of the system for which the session is to be started.

TRACE
starts a CSE data trace.

Usage Notes

1. This command is restricted to the PVM operator or an alternate operator specified on an AUTHORIZ record in the PVM configuration file.
2. The START GRAF command must be issued for each device being made available for users to DIAL.
3. When issued from another virtual machine using SMSG, you can use a command prefix with this command to tell PVM how to respond to the command. See "Using Command Prefixes" on page 255 for more information about command prefixes.

STATUS



Purpose

Use the STATUS command to determine the status of the system pools or the status of tasks within the system. You can use the output from the STATUS SYSTEM to determine if the sizes of the system pools are correct. The other options can be used for problem determination.

Operands

SYstem
displays information about the system Post Queue Element pool and buffer pool. This information gives the allocated size, peak usage, and current usage of each pool. To adjust these allocations, change the value specified on the BUFFERS record in the configuration file.

LINE *vaddr*
displays information concerning the associated line driver. This information includes the DVMTCB control block address, current task status, task flags, last (or current) CAW address, next trace entry address in the wrap trace table, and the CCW pointed to by the CAW. *vaddr* specifies the address of the line driver.

User *userid/devid*
displays information concerning the task associated with the specified ID. This information includes the DVMTCB control block address, current task status, and task flags.

The *userid/devid* identifies the user (*userid*) and the attached display or printer device (*devid*). The subject ID must be on the local system.

Appl *ldid*
displays information concerning the logical device support task associated with the specified ID. This information includes the DVMTCB control block address, current task status, and task flags.

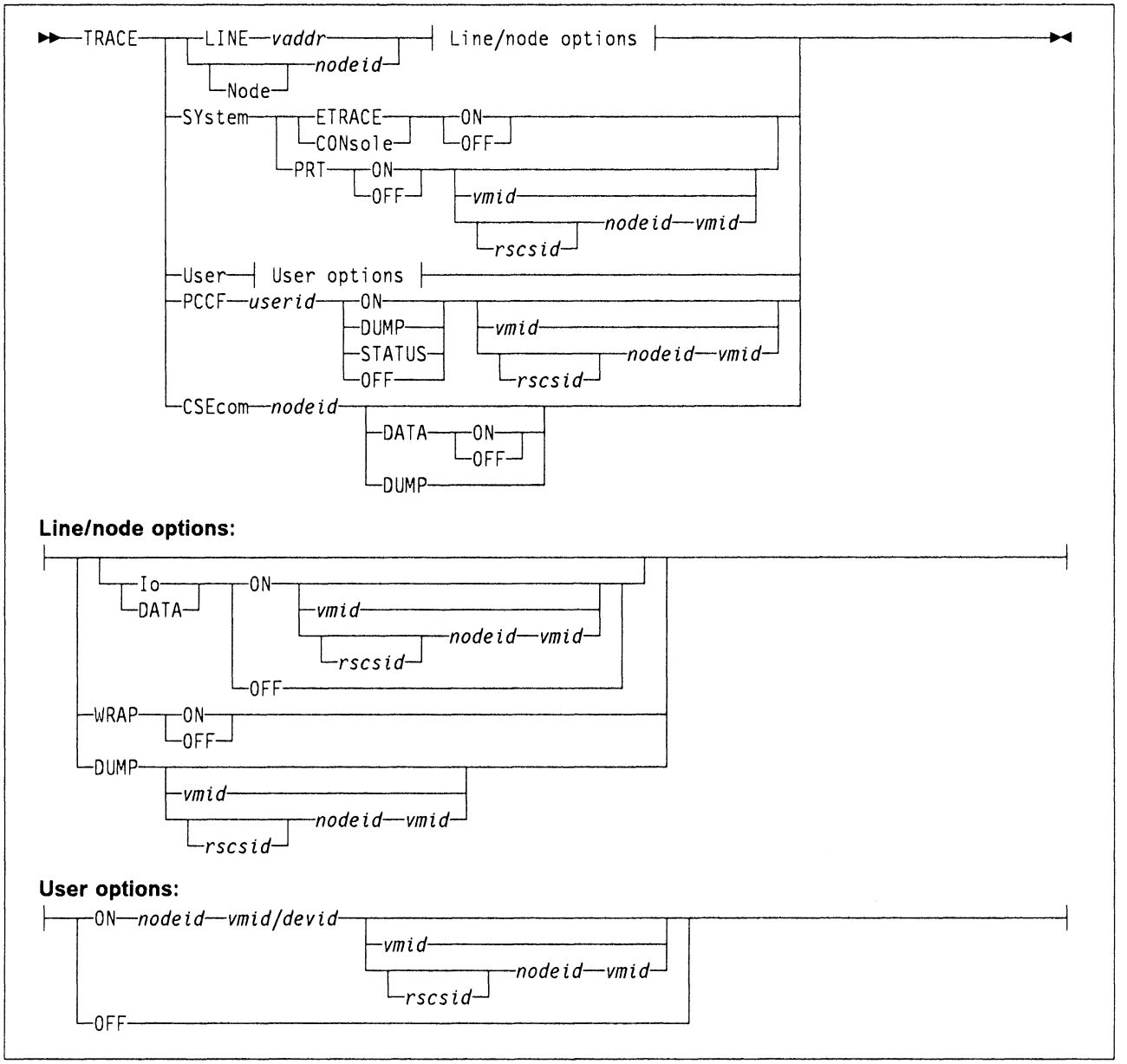
The *ldid* specifies the address of the logical device support task. The address should be specified in the form *Lnnn*.

Node *nodeid*
displays information specifies that information concerning the associated line driver whose node is *nodeid*. This information includes the DVMTCB control block address, current task status, task flags, last (or current) CAW address, next trace entry address in the wrap trace table, and the CCW pointed to by the CAW.

Usage Notes

1. This command is restricted to the PVM operator or an alternate operator specified on an AUTHORIZ record in the PVM configuration file.
2. When issued from another virtual machine using SMSG, you can use a command prefix with this command to tell PVM how to respond to the command. See "Using Command Prefixes" on page 255 for more information about command prefixes.

TRACE



Purpose

Use the TRACE command to start or stop the line tracing facility, the PVM multitasking supervisor trace facility, or the tracing of a PCCF support task.

Operands

LINE *vaddr*
specifies the virtual line address (*vaddr*) associated with the PVM line for which tracing is requested. The line address must be a valid address as specified by a LINK record or a DEFINE command.

Node *nodeid*
specifies the node name (*nodeid*) associated with the PVM line for which tracing is requested. The node name specified must be a valid node name as specified by a LINK record or a DEFINE command.

Io initiates a trace of line driver I/O activity for the specified line driver.

DATA
initiates a buffer trace for the specified line driver. Output goes to a dynamically defined virtual printer.

WRAP
causes wraparound I/O tracing for an attached line driver task. Output is stored in a buffer area managed by the affected task.

DUMP
causes the associated task to dump the local task control blocks and the internal I/O trace table.

SYstem
causes a trace for the PVM multitasking supervisor. Specifying OFF terminates only the specified system tracing option. Other system trace options are not terminated.

ETRACE
writes the output to a CPTRAP supported reader file. The trace records may be accessed later by IPCS.

Note: You can only use ETRACE with a VM/SP Release 6 or VM/ESA Release 1.0 (370 Feature) level of CP. You cannot use ETRACE with VM/XA or VM/ESA Release 1.0 (ESA Feature).

CONsole
writes the output to the PVM virtual console at virtual address 009.

PRT
writes the output to the virtual printer address 00F.

User
begins or ends tracing for a specific PVM session. When tracing is active, each task within PVM that is processing session data checks to see if the current piece of data is associated with the session to be traced. If it is, the CP DUMP command dumps the data block. The trace record contains the module name of the task producing the record, the task name, and the time stamp.

ON *nodeid vmid/devid*
specifies the identification of the node (*nodeid*) and PVM user (*vmid*) or attached display (*devid*) of the session to be traced. The session need not be currently active. Because a given session has both an origin node and user ID, and a destination node and user ID, you can specify either pair in the TRACE command.

If the *vmid* is the user ID of a MPVM user, the data for all of the user's sessions will be traced if the *nodeid* is the origin node ID of the MPVM user. If the *vmid* is a PVM user ID for a single session in the form of *ses*nnnn*, only data for that particular session is traced.

ON
initiates the trace.

OFF
terminates the trace.

PCCF *userid*
specifies the PCCF support task. The *userid* is the virtual machine for which PVM is handling PCCF connections.

STATUS
provides PCCF path information.

vmid
issues a CP SPOOL command to transfer output to the reader (*vmid*).

rscsid nodeid vmid
issues a CP SPOOL command and a CP TAG command to send the output over the RSCS network (*rscsid*) to the reader (*vmid*) at the specified node (*nodeid*).

CSEcom *nodeid*
traces a CSE communications task. The *nodeid* is the node name of the system for which the task is to be traced.

DATA
dumps CSE buffer data and traces DIAGNOSE X X'F0' information.

For TRACE LINE, TRACE NODE, and TRACE USER, if you do not specify *rscsid*, it defaults to RSCS.

Usage Notes

1. This command is restricted to the PVM operator or an alternate operator specified on an AUTHORIZ record in the PVM configuration file.
2. All TRACE operands are initially set off.
3. The following table lists the file naming conventions for trace output files.

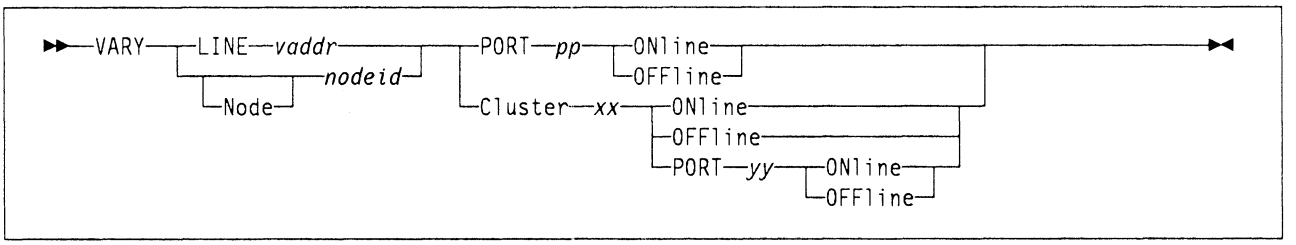
TRACE Command Options	Filename	Filetype
IO	*link_name*	IOTRACE
DATA	*link_name*	DATATRAC
DUMP	*link_name*	TASKDUMP
SYstem PRT	*local_node_ID*	SUPTRACE
User	*local_node_ID*	SESSTRAC

   The file naming convention for trace output with the WRAP option and for trace output with the SYSTEM ETRACE option and SYSTEM CONSOLE option is not applicable.

4. The following CSE events are traced:
   - Returning a copy of the CSE tables
   - Returning the number of SPFBKs
   - Synchronizing with another system
   - Reading queue 1
   - Processing a buffer from *nodeid*
   - Posting a spool file synchronization process
   - Flagging *nodeid* inactive
   - Dropping a communication link.

5. When issued from another virtual machine using SMSG, you can use a command prefix with this command to tell PVM how to respond to the command. See "Using Command Prefixes" on page 255 for more information about command prefixes.

VARY



Purpose

Use the VARY command to alter the availability status of links and devices. Once a port or control unit is varied off-line, the system does not start any new sessions on that cluster or port. Once all existing sessions on that cluster or port have ended, the system does not communicate with the control unit or port.

Operands

LINE *vaddr*
identifies that line *vaddr* is varied on-line or off-line.

Node *nodeid*
identifies that node *nodeid* is varied on-line or off-line.

PORT *pp*
identifies that port *pp* is varied on-line or off-line.

ONline
brings port *pp* on-line.

OFFline
sets port *pp* off-line.

CLuster *xx*
indicates that cluster *xx* is varied on-line or off-line. *xx* is the cluster number. It must be defined by a CLUSTER record in the configuration file.

ONline
brings the associated cluster on-line. If the cluster is currently in an error recovery wait period, the wait period is terminated, and the cluster is immediately rechecked for availability.

OFFline
sets the associated cluster off-line. All ports not in session are deactivated. As each session with a port on the cluster terminates, that port is deactivated. When all the ports on the cluster are deactivated, the cluster is placed off-line and is no longer polled.

PORT *yy*
indicates that port *yy* on the associated cluster is varied on-line or off-line. If the port is in session, it is brought off-line when the session terminates.

ONline
brings port *yy* on-line if it is available.

OFFline
sets port *yy* off-line if it is not in session.

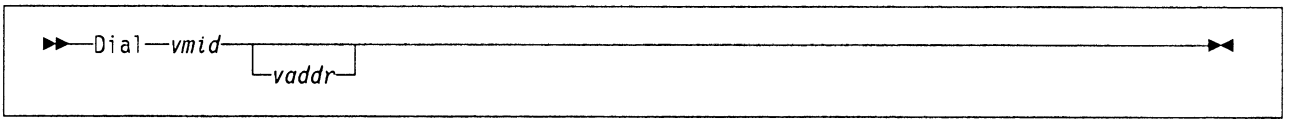
Usage Notes

1. This command is restricted to the PVM operator or an alternate operator specified on an AUTHORIZ record in the PVM configuration file.
2. When issued from another virtual machine using SMSG, you can use a command prefix with this command to tell PVM how to respond to the command. See "Using Command Prefixes" on page 255 for more information about command prefixes.

CP Commands Used with PVM

This section describes CP commands that you can use for interaction with PVM.

DIAL



Purpose

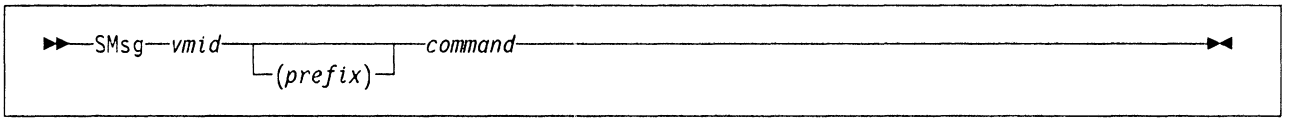
Use the CP DIAL command to access PVM directly from a VM logon screen. The PROFILE PVM or an authorized user must issue a START GRAF command for each device being made available for users to DIAL.

Operands

vmid
is the virtual machine identification of the PVM user ID.

vaddr
is the virtual display device address.

SMSG



Purpose

Use the CP SMSG command to issue PVM commands to PVM from another virtual machine.

Operands

vmid
is the virtual machine identification of the PVM user ID.

(*prefix***)**
is any valid PVM command prefix. See"Using Command Prefixes" on page 255 for more information about command prefixes.

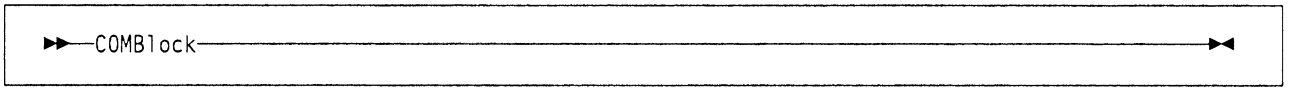
command
is any valid PVM command.

PVM/IPCS Support Commands

The commands added to the IPCS command set for use on PVM problems are valid only if the optional PVM/IPCS support is installed. These commands are generally used by the persons responsible for reporting system problems and performing problem diagnostics. The PVM/IPCS support commands are the IPCSSCAN, IPCSDUMP, and MAP commands. Included in the support of the IPCSSCAN command are these subcommands:

- COMBLOCK
- DEVQUE
- HELP
- LINKS
- ROUTES
- SYSBLOCK
- TASKQUE
- USERQUE.

COMBLOCK



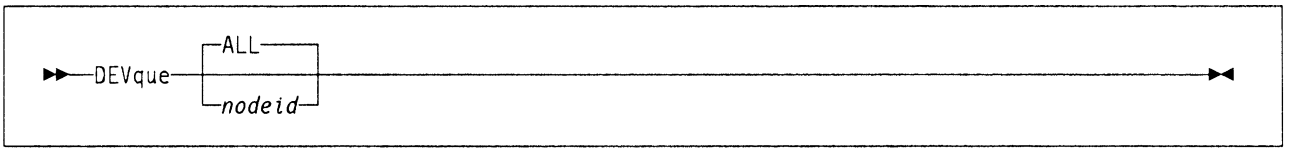
Purpose

Use the COMBLOCK subcommand to display the DVMCOMN control block from the PVM system dump.

Usage Notes

1. This command is valid only in IPCSSCAN mode.
2. See "DVMCOMN (PVM Multitasker Common Area) Format" on page 427 for a description of the DVMCOMN format.

DEVQUE



Purpose

Use the DEVQUE subcommand to display the active logical device support task descriptor blocks (DVMAATs) from the list associated with a particular node (LINK) or for all nodes defined to the PVM system. The information is formatted for display.

Operands

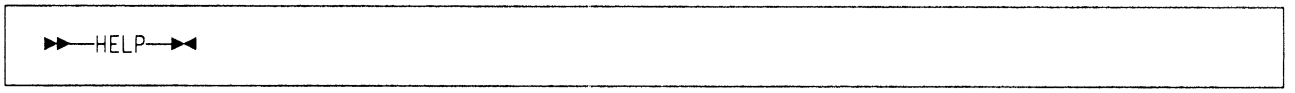
nodeid
specifies the node ID of the link for which the associated device queue is to be displayed.

ALL
specifies that the device table for every link is to be displayed.

Usage Notes

1. This command is valid only in IPCSSCAN mode.
2. See "DVMAAT (Active Application Task Descriptor) Format" on page 419 for the DVMAAT format.

HELP



Purpose

Use the HELP subcommand to display a list of the available common and PVM/IPCS IPCSSCAN subcommands and their explanations.

Usage Notes

1. This command is valid only in IPCSSCAN mode.

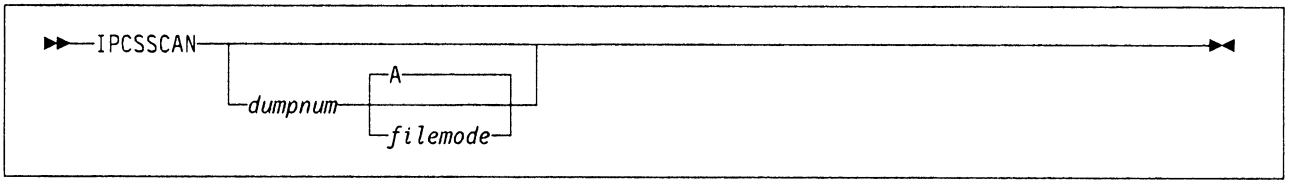
IPCSDUMP

Purpose

Use the IPCSDUMP command to create a problem report and attach the compressed load map to the dump. The compressed load map must be attached to the dump before IPCSSCAN can be used. The compressed load map is created with the MAP command.

For more information about the IPCSDUMP command, see the *VM/ESA: Interactive Problem Control System Guide and Reference for 370*.

IPCSSCAN



Purpose

Use the IPCSSCAN command to start the processing that allows you to scan through a PVM system dump. Once in IPCSSCAN mode, you can select the areas to be viewed by using IPCSSCAN subcommands. These may be any IPCS subcommands or any of the PVM/IPCS subcommands:

- COMBLOCK
- DEVQUE
- HELP
- LINKS
- ROUTES
- SYSBLOCK
- TASKQUE
- USERQUE.

For more information about the IPCSSCAN command, see *VM/ESA: Interactive Problem Control System Guide and Reference for 370*.

Operands

dumpnum
is the dump number that is contained in the PRBnnnnn number of the dump name.

filemode
is the filemode of the disk.

A specifies a filemode of A.

Usage Notes

1. If you issue the IPCSSCAN command with no parameters, you are prompted for the dump number that is contained in the PRBnnnnn number of the dump name and the file mode of the disk. If file mode is not specified, A is the default.

LINKS

Purpose

Use the LINKS subcommand to display the link descriptor block (DVMALT) for a particular node (LINK) or for all nodes defined to the PVM system. The information is formatted for display.

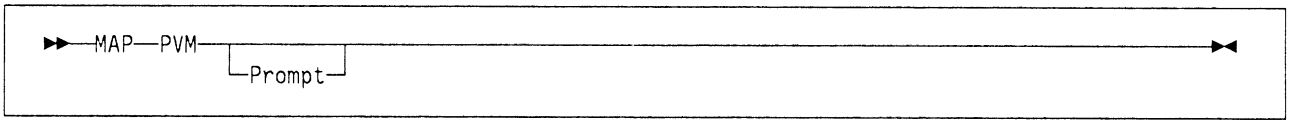
nodeid
specifies the node ID of the link for which the associated link list entry is to be displayed.

ALL
specifies that the link list entry for every link is to be displayed.

Usage Notes

1. This command is valid only in IPCSSCAN mode.
2. See "DVMALT (Active Link Table Entry) Format" on page 420 for the DVMALT format.

MAP



Purpose

Use the MAP command to compress the CMS and PVM load maps into one compressed load map. The final compressed load map for PVM will be the file PVMIPCS MAP A1. The default names of the input maps are CMSNUC MAP A1 and PVM MAP A1 for the CMS and PVM maps respectively.

Operands

PVM
indicates request is for a PVM map. The map produced is disk-resident and usable by IPCSDUMP.

Prompt
specifies that the user does not want to use the assigned default names for input. Prompt messages are issued for each file name required.

The acceptable reply to these messages is:

```
filename filetype filemode
```

If not specified, `filetype` and `filemode` defaults to `MAP A1`. It is also possible to submit one of four other replies to the prompt message as follows:

CMS Enter CMS subset

NULL Use the assigned default name for this file.

QUIT Terminate the MAP command.

NONE Bypass processing this input map. This option is provided for secondary input maps only and will be ignored for other files.

Examples

To create a compressed load map using the **PROMPT** option:

```
  Enter:  map pvm p
System:

DTVMAP819R  ENTER 'FN' OF THE PVM IPCS MAP TO BE
            CREATED, OR ENTER A NULL LINE

  Enter:  pvmipcs
System:

DTVMAP806R  FOR PVM, ENTER FN FT FM OF THE NUCLEUS MAP,
            OR ENTER A NULL LINE, CMS, NONE, OR QUIT
```

Enter: `cmsnuc map a`
System:

```
DTVMAP808I   PVM IPCS MAP PVMIPCS MAP A1 CREATED FROM CMSNUC MAP A1
DTVMAP806R   FOR PVM, ENTER FN FT FM OF THE PVM MAP,
             OR ENTER A NULL LINE, CMS, NONE, OR QUIT
```

Enter: `pvm map a`
System:

```
DTVMAP802I   PVM MAP PVM MAP A1 ADDED TO IPCS MAP
```

This completes the building of the IPCS MAP.

ROUTES

Purpose

Use the ROUTES subcommand to display the route table that was in use by the PVM system.

Usage Notes

1. This command is valid only in IPCSSCAN mode.
2. See "DVMROUT (Entry in Route List) Format" on page 430 for the DVMROUT format.

SYSBLOCK

Purpose

Use the SYSBLOCK subcommand to display the DVMSYSB control block from the PVM system dump.

Usage Notes

1. This command is valid only in IPCSSCAN mode.
2. See "DVMSYSB (PVM Multitasker System Control Block) Format" on page 431 for the DVMSYSB format.

TASKQUE

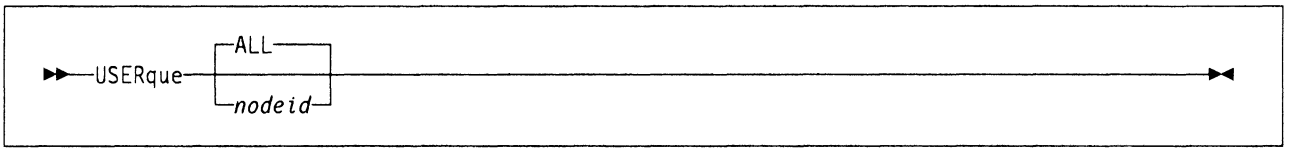
Purpose

Use the TASKQUE subcommand to display the task control blocks (DVMTCBs) of those tasks on the PVM system's run queue.

Usage Notes

1. This command is valid only in IPCSSCAN mode.
2. See "DVMTCB" on page 434 for the DVMTCB and DVMNTCB formats.

USERQUE



Purpose

Use the USERQUE subcommand to display the active user support task descriptor blocks (DVMAUTs) from the list associated with a particular node (LINK) or for all nodes defined to the PVM system. Also, USERQUE can be used to generate a (DVMAPT) list from the dump. The information is formatted for display. DVMCPOR and DVMACT control blocks are accessed to locate DVMAPT.

Operands

nodeid
specifies the node ID of the link for which the associated device queue is to be displayed.

ALL
specifies that the user list for every link is to be displayed.

Usage Notes

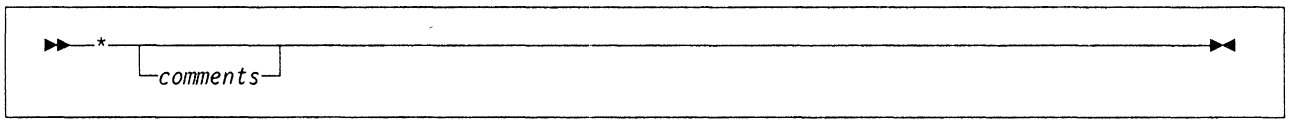
1. This command is valid only in IPCSSCAN mode.
2. See "DVMAUT (Active User Interface Link Descriptor) Format" on page 423, "DVMAPT (Active Remote Port Descriptor) Format" on page 422, "DVMCPOR (3271-4 Cluster Port Entry) Format" on page 429, "DVMACT (Cluster Table Entry) Format" on page 420 for more information about the respective control blocks.

Appendix D. PVM Configuration File Records

PVM configuration files contain fixed-length, 80-character records. Input must be in uppercase and is restricted to columns 1 through 71 with no continuation. Entries in a configuration file record must be separated by one or more blanks.

For more information on using PVM configuration file records, see "Creating the Configuration File" on page 66, and "Storage Considerations" on page 25.

* (Comment)



Purpose

Use the * record to imbed comments in the configuration file. You can place this record anywhere in the file. The record is ignored during configuration file processing.

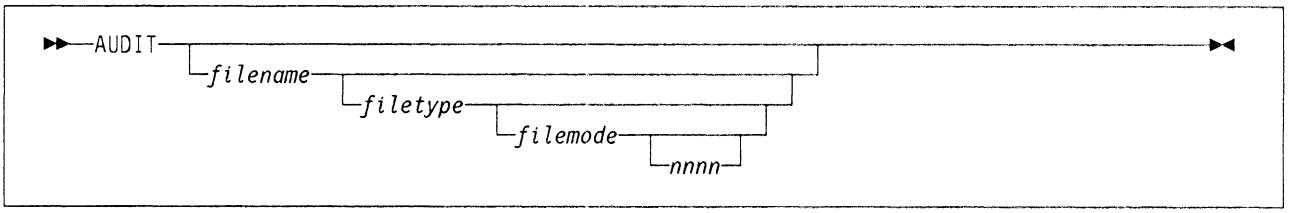
Operands

comments
is any text message. This information, plus at least one blank, must follow the asterisk (*).

Usage Notes

1. The Comment record must begin in column one of the PVMG configuration file.
2. There can be no blank lines in the PVM configuration file. To insert lines between records, use this record.

AUDIT



Purpose

Use the AUDIT record to indicate that console data is to be stored in a specified CMS file. This file is created the first time data is written to it. If the Message Service Facility is enabled, messages resulting from CMS commands or commands submitted directly to CP are stored. Data is stored chronologically, along with the current date and time. When the audit file is full, it wraps to the beginning of the file, overlaying the existing records.

© Copyright IBM Corp. 1991,1993

The audit file is a fixed length file, with the first record containing the record number of the next record to be written. The length of the AUDIT file depends on if the audit records contain message numbers and/or time stamps. The length of the AUDIT file is:

- 129 bytes if the audit record contains neither time stamps nor message numbers
- 140 bytes if the audit record does not contain time stamps but does contain message numbers
- 129 bytes + *maximum timestamp length* + 1, if the audit record contains a time stamp but does not contain message numbers
- 140 bytes + *maximum timestamp length* + 1, if the audit record contains a time stamp and message numbers.

The + *1* allows for a single blank following the timestamp.

Maximum timestamp length refers to the longer of the AM or the PM timestamp for the default language. However, if the default language is a DBCS language, the *maximum timestamp length* is the longest of the:

- AM or PM timestamp for the default language
- AM or PM timestamp for the backup language.

All LOCAL, LINK, PORT, CLUSTER, CLPORT, GROUP, ROUTE, or AUTHORIZ records must precede this record. If you omit this record, no audit file is created. If you enter more than one AUDIT record in the configuration file, only the last AUDIT record in the list is accepted. PVM ignores the other AUDIT records.

Operands

filename
is the file name of the CMS file containing the console data. If *filename* is not specified, the default value is CONSOLE.

filetype
is the file type of the CMS file containing the console data. If *filetype* is not specified, the default value is LOG.

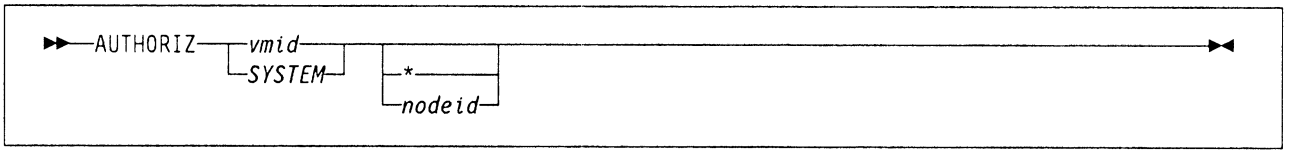
filemode
is the file mode of the CMS file containing the console data. If *filemode* is not specified, the default value is A6.

nnnn
is the maximum size of the audit file in 1K records. The value of *nnnn* ranges from 1 to 1000, allowing a maximum of 1 million records. The default is 999,999 records.

Usage Notes

1. You should specify a maximum size for the audit file that prevents the PVM minidisk from getting full.

AUTHORIZ



Purpose

Use the AUTHORIZ record to identify users who are authorized to issue the restricted subset of PVM commands and the nodes from which these commands may be issued.

All LOCAL, LINK, PORT, CLUSTER, CLPORT, GROUP, or ROUTE records must precede this record.

Operands

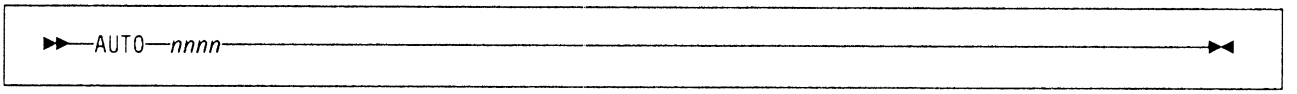
vmid
is the identification of a VM user (or virtual machine) authorized to issue restricted PVM commands. Without specifying * or *nodeid*, this authorization is for the local node only.

SYSTEM
authorizes the PVM operator to issue commands to nodes (* or *nodeid*) other than the local node.

*
authorizes the user to issue restricted PVM commands *from any node.*

nodeid
is a specific node where *vmid* is authorized to issue restricted PVM commands.

AUTO



Purpose

GUPI

Use the AUTO record to indicate that PVM is to automatically execute the AUTO PVM exec if it exists. AUTO PVM is a customer written exec that can issue CP, CMS, or PVM commands to monitor activities such as status of links or nodes in the PVM network. PVM does not pass the AUTO PVM exec any inputs, nor does it take action on any passed return codes or data.

The exec is executed based on the time interval specified in the AUTO record. If the AUTO record is not specified, no automatic execution is performed.

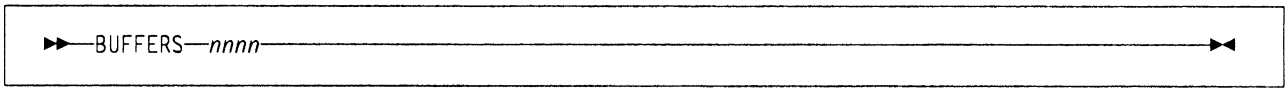
GUPI end

All LOCAL, LINK, PORT, CLUSTER, CLPORT, GROUP, ROUTE, or AUTHORIZ records must precede this record.

Operands

nnnn
is the number of minutes between invocations of the exec. The value of *nnnn* ranges from 1 to 1440.

BUFFERS



Purpose

Use the BUFFERS record to indicate the number of 4096-byte pages allocated for internal data buffers. PVM also uses the BUFFERS record to compute the size of the Post Queue Element (PQE) pool.

All LOCAL, LINK, PORT, CLUSTER, CLPORT, GROUP, ROUTE, or AUTHORIZ records must precede this record. If you omit this record, the default value is 30 pages. If you enter more than one BUFFERS record in the configuration file, only the last BUFFERS record in the list is accepted. PVM ignores the other BUFFERS records.

Operands

nnnn
is the number of pages to be allocated in the range 25 to 2048. The number of pages specified should be greater than half the maximum number of users. Specifying insufficient buffers can degrade system performance, or cause PVM to completely quiesce.

Usage Notes

1. To calculate the *nnnn* value, use the following formulas:

When (SMAX/2 + 15) is greater than UMAX:

```
BUFFERS = 25 + 3(NMAX) + CMAX + 2(EMAX) + BMAX + SMAX/2
```

Or, when (SMAX/2 + 15) is less than or equal to UMAX:

```
BUFFERS = 10 + 3(NMAX) + CMAX + 2(EMAX) + BMAX + UMAX
```

NMAX
is the maximum number of active PVM to PVM network CTCA and APPC type line drivers, and PVM to PVMG IUCV line drivers.

CMAX
is the maximum number of active 3088 line drivers times the average number of buffer pages per driver. The number of pages is specified by the BLK operand of the LINK configuration file record. If BLK is not specified, the default number of pages is 6 (3 for input and 3 for output).

EMAX
is the maximum number of active 327x line drivers.

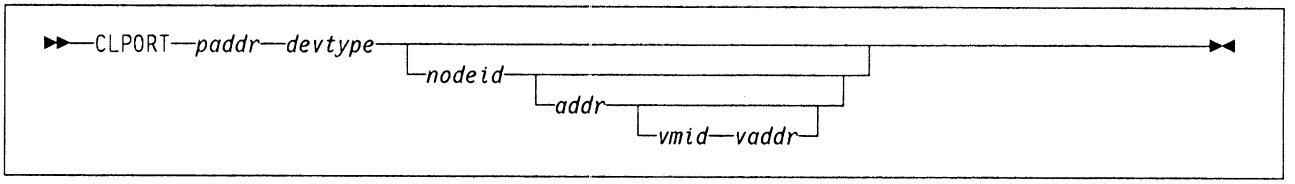
BMAX
is the maximum number of active R3270, ROCF, and BSCA line drivers.

SMAX
is the maximum number of active PVM user and PCCF sessions, plus the number of MPVM user sessions, active at any one time.

UMAX
is the maximum number of sessions simultaneously accessing the local CP system as a target node.

2. The calculation for BUFFERS is only an approximation. The PVM operator should monitor the usage of the buffer pool using the STATUS SYSTEM command. For larger systems, the TOTAL BUFFER PAGES displayed should exceed the MAX USED by 50 pages.

CLPORT



Purpose

Use the CLPORT record to define a port on a remote 3271 or 3274 control unit driven by the PVM R3270 line driver.

The CLPORT record must follow the associated CLUSTER record that defines the control unit the device is attached to. CLPORT records for a specific cluster may be in any order. Any ports that are not defined by a CLPORT record are not used by the line driver.

Operands

paddr
is the device (port) address. The value of *paddr* ranges from 0 to 31.

devtype
indicates the device type. Valid types are:

3277	3278-5	3286
3278-2	3279-2	3287
3278-3	3279-3	3288
3278-4	3284	3289

The following parameters apply only to printer type devices. If the specified device is a printer and none of the following parameters are entered, the printer is not used by PVM.

nodeid
is the identification of the node the printer is connected to. The node specified can be either a 3270 emulator node or a VM system node. If the

node is a 3270 emulator, it must have a printer port defined. If the node is a VM system, a logical printer is created; the VM system must have the CP extensions to the logical device support to create logical printers.

addr
defines a specific port address when *nodeid* is a 3270 emulator node or defines a logical device offset when *nodeid* is a VM system node. The offset (*addr*) is used in creating a logical device address.

To create the logical device address, *addr* is added to the logical device base address for PVM. The offset is in the form L*xxx*, where *xxx* is a hexadecimal value between 000 and FFF. Some VM/XA systems may use a different hexadecimal value for this range.

If L000 is specified, the next available logical printer device address in the PVM server machine will be used for the printer session. Therefore, logical device address 000 cannot be requested.

vmid vaddr
is the identification of the virtual machine and the virtual address to which the logical printer is attached to when it is created.

Examples

1. To attach a remote printer to a PVM system use the following form of the CLPORT record:

   CLPORT *paddr devtype nodeid addr vmid vaddr*

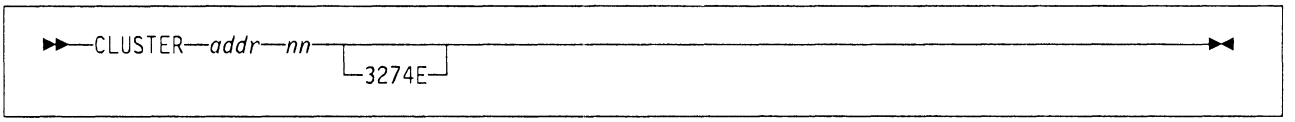
   This defines a printer, allows a logical device to be created on a CP system (*nodeid*) representing the printer, assigns an offset value for the logical device address (*addr*), and causes PVM to attach the logical device to a virtual machine (*vmid*) at a given address (*vaddr*). The ATTACH must be issued from a Class B machine.

2. To specify that a remote printer should be attached to a non-PVM node use this form of the CLPORT record:

   CLPORT *paddr devtype nodeid addr*

   This defines a printer and causes connection to a specific port (*addr*) on an emulator link (non-PVM *nodeid*). If a port number is not specified connection would be made to the next available port.

CLUSTER



Purpose

Use the CLUSTER record to define a remote 3271 or a remote 3274 control unit on a TP line driven by the PVM R3270 line driver. There can be up to 16 CLUSTER records for each R3270 line.

The CLUSTER record must follow the LINK record defining the remote line the control unit is connected to or the last CLPORT record defining the ports for a previous cluster on the same line.

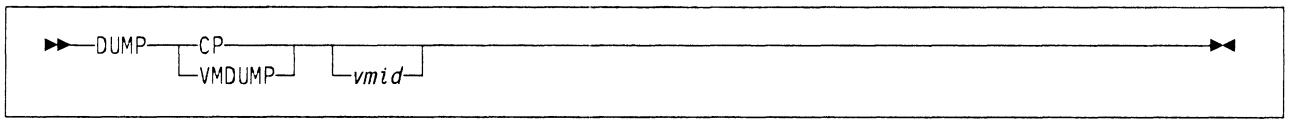
Operands

addr
is the remote 3270 control unit address. The *addr* is a two-digit number ranging from 0 to 31.

nn is the maximum number of ports supported by this control unit. The value of *nn* ranges from 1 to 32. This field also sets the range of port addresses specified by the succeeding CLPORT records. If *nn* is set to 16, only devices 0 through 15 are defined by CLPORT records.

3274E
indicates that the control unit is a 3274/3271/3174 that supports 3270 extended data streams. If this option is omitted, PVM assumes that the cluster does not support extended data streams.

DUMP



Purpose

Use the DUMP record to indicate whether the dump taken during abnormal termination of PVM is a CP dump or a VM IPCS dump.

All LOCAL, LINK, PORT, CLUSTER, CLPORT, GROUP, ROUTE, or AUTHORIZ records must precede this record. If you omit the DUMP record, the default is CP-type dump. If you enter more than one DUMP record in the configuration file, only the last DUMP record in the list is accepted. PVM ignores the other DUMP records.

Operands

CP
indicates that on abnormal termination of PVM the dump is CP DUMP.

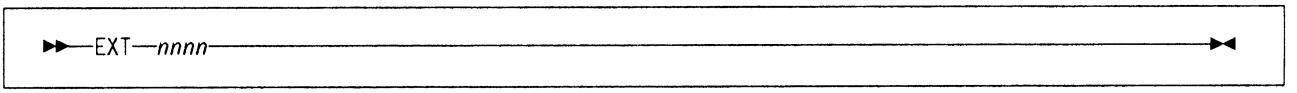
VMDUMP
indicates that on abnormal termination of PVM the dump is VMDUMP.

vmid
indicates where the dump is to be spooled. The default is written to the printer for a CP dump and spooled to the *vmid* of IPCS for a VMDUMP.

Usage Notes

1. The PVM/IPCS Extension and the VM/ESA and VM/XA Dump Viewing facility require a VMDUMP type dump. Therefore, VMDUMP *must* be specified if you use the PVM/IPCS or the VM/ESA and VM/XA Dump Viewing facility.

EXT



Purpose

Use the EXT record to specify the maximum number of entries allotted in the internal table that handles VMCF and logical device external interrupts and also interrupts on both IUCV and APPC/VM paths. An entry is made for a user accessing this PVM system from the CMS environment using the PASSTHRU command, and two entries are made for a user using the MPVM command. One entry is made for logical devices created for this PVM node. Add two entries for use by the system. Specify enough entries to handle the peak number of simultaneous CMS PASSTHRU and MPVM users, logical devices, and IUCV and APPC type links.

All LOCAL, LINK, PORT, CLUSTER, CLPORT, GROUP, ROUTE, or AUTHORIZ records must precede this record. If you omit this record, the default value is 50 entries. If you enter more than one EXT record in the configuration file, only the last EXT record in the list is accepted. PVM ignores the other EXT records.

Operands

nnnn
is the maximum number of entries created. The value of *nnnn* ranges from 10 to 4096.

Usage Notes

1. To calculate the value for EXT use the following formula:

```
EXT = 3 + VMAX + 2(AMAX) + UMAX +CMAX + 2(MPVM) + IMAX
```

VMAX
is the maximum number of users simultaneously accessing PVM by the PASSTHRU command.

AMAX
is the maximum number of active APPC/VM line drivers.

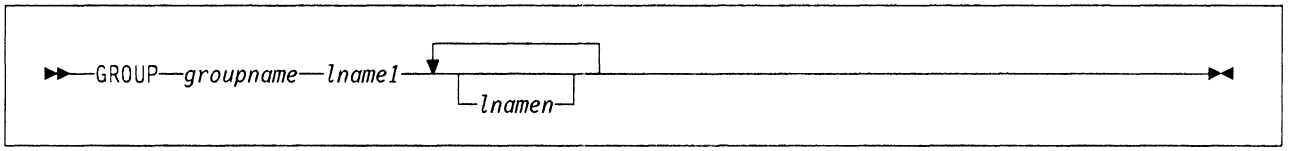
UMAX
is the maximum number of users simultaneously accessing the local CP system as a target node.

CMAX
is the maximum number of simultaneous PCCF sessions.

MPVM
is the maximum number of simultaneous MPVM users.

IMAX
is the maximum number of IUCV line drivers going to PVMG virtual machine(s).

GROUP



Purpose

Use the GROUP record to define a set of links that form a line group. The group name is the node name of the adjacent PVM system the links are connected to. The same group name may appear on more than one group record. In this case, the line group consists of the sum of all the links specified on the multiple group records.

The GROUP records must be placed after all the LOCAL, LINK, PORT, CLUSTER, and CLPORT records, but the GROUP records must precede any other records.

The only types of links that can belong to a line group are BSCA, CTCA, APPC, and 3088 links. An APPC link cannot appear in a GROUP with another APPC link. However, it may be combined with any of the other types of supported links.

The GROUP record must contain at least one link node name. In a group of two or more links, the link types do not have to be identical. When a link is defined as part of a group, the link node name **should not** appear on a SELECT record.

Note: It is **very** important that *groupname* specified in the GROUP record be a name that is unique throughout the **entire** PVM network. This includes the R3270 links. Because *groupname* is used as a node name on the PVM Selection Menu, the location of the site should be apparent by the name.

Operands

groupname
is the node name of the line group.

lname1
is the link node name in the line group.

lnamen
is the link node name in the line group.

I/O

Purpose

Use the I/O record to specify the maximum number of entries allotted in the PVM internal table that handles I/O devices. Use one entry for each virtual address that PVM performs I/O operations on. Therefore, there will be one entry for each active line driver (START LINE), one for each attached display support task (START GRAF), one for each attached printer support task (START PRT), and one for the PVM user ID display station. You should specify enough entries to handle the peak number of active line drivers and attached display support tasks.

All LOCAL, LINK, PORT, CLUSTER, CLPORT, GROUP, ROUTE, or AUTHORIZ records must precede this record. If you omit this record, the default value is 50. If you enter more than one I/O record in the configuration file, only the last I/O record in the list is accepted. PVM ignores the other I/O records.

Operands

nnnn
is the maximum number of I/O devices expected. The value of *nnnn* ranges from 10 to 2048.

Usage Notes

1. To Calculate the value for I/O use the following formula:

```
I/O = 1 + LMAX + GMAX + PMAX
```

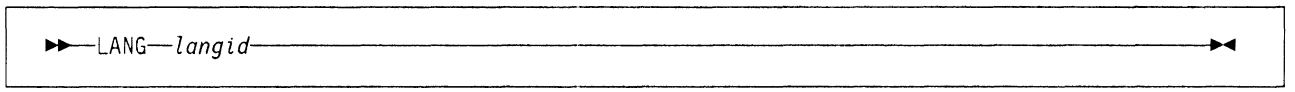
where:

LMAX
is the maximum number of active line drivers. Add one extra for each active 3088 type line driver.

GMAX
is the maximum number of attached display support tasks.

PMAX
is the maximum number of attached printer support tasks.

LANG



Purpose

Use the LANG record to identify a particular language as the default national language for a node.

During PVM initialization, the backup language is considered the national language for a node before the LANG record is processed. It continues as the default national language for the node if the LANG configuration file record is not specified.

You can place the LANG record anywhere within the configuration file; it takes effect immediately. The LANG record can be entered before the LOCAL record to establish the language used for any error messages issued in conjunction with subsequent configuration file processing.

Operands

langid
is the identification of the selected language. It is a 1-to-5-character language identifier.

The IBM-provided languages are:

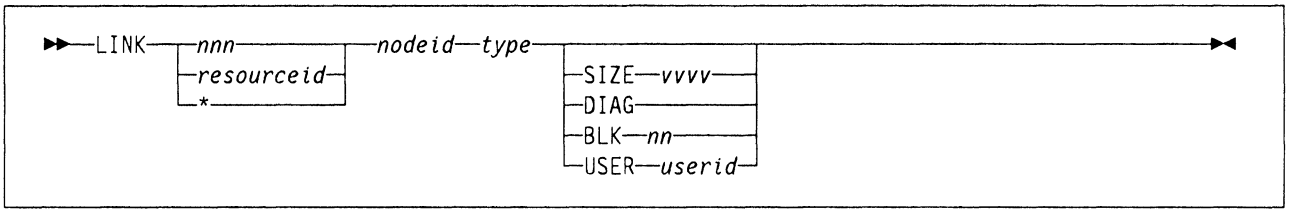
AMENG American English

KANJI Japanese

UCENG Uppercase English

GER German

LINK



Purpose

Use the LINK record to define the line address and line type (where applicable) for connecting PVM to a remote system, including PVMG.

The LOCAL record must precede this record. The order of LINK records in the configuration file should correspond to their frequency of use. If the line type is 327X, the LINK record must be followed by PORT records defining all active ports. If the line type is R3270, the LINK record must be followed by CLUSTER and CLPORT records defining all active clusters and ports. The same node identifier cannot appear in more than one LINK record.

Note: It is **very** important that the *nodeid* specified in the LINK record be a name that is unique throughout the **entire** PVM network. This includes the R3270 links.

Operands

nnn
is the line address. This address ranges from X'20' to X'FF' for BSCA, 327X, ROCF, and R3270 type links, and from X'100' to X'FFF' for CTCA and 3088 links.

resourceid
is the name of the resource nickname to which you are connecting. The *resourceid* is a 1-to-8-character string. This value applies only to APPC type links. The *resourceid* refers to the nickname entry found in the CMS communications directory (UCOMDIR NAMES). PVM uses the *resourceid* to invoke a CMSIUCV RESOLVE function.

* indicates that a line address is not provided. For a type of IUCV, this parameter is required.

nodeid
is the node name of the system on the remote end of this link if this link does not belong to a line group. If this link is part of a line group, the node name cannot be the same name as any PVM network node. Node names are available to the user, through the PVM Selection Menu, for node selection at session initiation.

NOTE: Node names for links defined as R3270 or individual links of a line group do not appear on the PVM Selection Menu.

type
specifies the type of line driver used to drive the line. The valid types are:

APPC uses the APPC/VM protocol to communicate over the logical path.

Note: You can only use APPC with VM/ESA or VM/SP Release 6

IUCV uses the IUCV protocol to communicate with a PVMG virtual machine.

3088 specifies that this link to another PVM is made by a 3088 Multisystem Channel Communications Unit, or by a ESCON channel. The 3088 driver supports a consecutive EVEN/ODD pair of 3088 addresses. Because the 3088 driver uses virtual addresses instead of real addresses, the real addresses do not necessarily have to be consecutive. However, to eliminate confusion the real addresses should be consecutive too.

Use this link type for communications with an channel-attached IBM 8232 LAN Channel Station. Use an ODD address on the LINK record used on the host side and an EVEN address on the IBM 8232 LAN Channel Station side.

The address specified in the LINK record indicates the address of the OUTPUT link. It must be the EVEN address on one side of the link and the ODD address on the opposite side of the link. If the specified address is EVEN, PVM assumes the input address is one higher. If the address is ODD, PVM assumes the input address is one lower.

Note: OPTION BMX (if supported in your VM environment) **MUST** be specified in the directory for this line driver to function, because the driver will start an I/O to both addresses at the same time.

CTCA specifies that this link to another PVM is made by a channel-to-channel adapter.

BSCA specifies that this link to another PVM system is made by a binary synchronous communications line.

327X specifies that a 3271/3274 emulator line driver is to be used. If 327X is specified, associated PORT records must immediately follow this record.

Note: This line must have either the Station Selection feature on 207X, or the Tributary option on 370X or the 4321/4331's Communication Adapter to function properly.

ROCF specifies that an ROCF line driver is to be used to communicate with a remote 4300 processor over a switched, voice-grade telephone line.

R3270 specifies that this is a remote 3270 cluster line, and the PVM remote 3270 line driver is to be used. The line connection may be switched point-to-point, nonswitched point-to-point, or nonswitched multipoint. If you specify an R3270 link, the LINK record must be followed by CLUSTER and CLPORT records defining all active clusters and ports. Otherwise, the system will ignore the R3270 LINK record.

SIZE *vvvv*

indicates the *threshold block size* for this link. This value governs the minimum block size to be transmitted by network and IUCV line drivers when there is a queue of data to send. The threshold block size should be set at a value that is low enough to reduce interference between interactive and bulk, but large enough to efficiently transmit the data. Generally, the value should not be greater than the number of characters that can be transmitted across the link in one second. For example, if the line is a 9600 BAUD TP link, 1200 characters can be transmitted per second.

The default value is 4096 for IUCV, CTCA, APPC, and 3088, and 1000 for any other types. The value of *vvvv* ranges from 200 to 4096.

DIAG
issues a DIAGNOSE code X'98' (REAL I/O) for this link. DIAG is only valid if the link type is 3088. For this option to be enabled, the VM system must support DIAGNOSE code X'98' and OPTION DIAG98 must be specified in the PVM directory. DIAG should not be specified for virtual 3088 to 3088 communication. DIAGNOSE code X'98' locks I/O buffers in real storage.

BLK *nn*
indicates the number of 4K areas that are tied together for the output transmission buffer. The value of *nn* ranges from 2 to 16. The default value is 3. BLK is only valid if the link type is 3088. The BUFFERS value may have to be increased depending on the number of 3088 links and the BLK values. PVM allocates from the buffer pool the number of pages specified in the BLK record on each side for every 3088 link. Therefore, the default is 6 pages for each 3088 link that is started. The 3088 line driver sends handshaking information to the other side that specifies an output size based on the BLK value and a maximum input size of 64K. When it reads the handshake from the other side, it adjusts the input size to fit the opposite sides output size. If the opposite side indicates a maximum receive size that is smaller than the BLK value, the output size is adjusted downward.

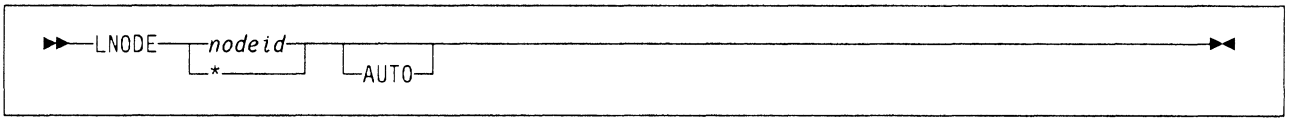
NOTE: This is a reserved capability. PVM will not send an input size which is less than the maximum size possible.

USER *userid*
identifies the local user that is requesting a connection to a remote PVM node over an APPC link, or the user ID of the connecting PVMG virtual machine. This *userid* (and password) must match the name defined in the :userid. field of the CMS communications directory. This *userid* is a 1-to-8-character string identifying the:

- Local user that is requesting a connection to a remote PVM node over an APPC link,
- PVMG virtual machine to which an IUCV connection is to be made.

This operand applies to an APPC type link and to an IUCV connection to a PVMG virtual machine, and it must be specified.

LNODE



Purpose

Use the LNODE record to define the node name that is selected by the ENTER key from the Initial Menu.

The AUTO option on this record allows you to bypass the Initial Menu and the PVM Selection Screen. If you want to go to a different system than the one

specified by the AUTO option on the LNODE record, you must log on to a CMS virtual machine and specify the target system on the CMS PASSTHRU command.

Note: This record does not apply to MPVM.

All LOCAL, LINK, PORT, CLUSTER, CLPORT, GROUP, ROUTE, or AUTHORIZ records must precede this record. If you omit this record, the default value is the node name from the LOCAL record.

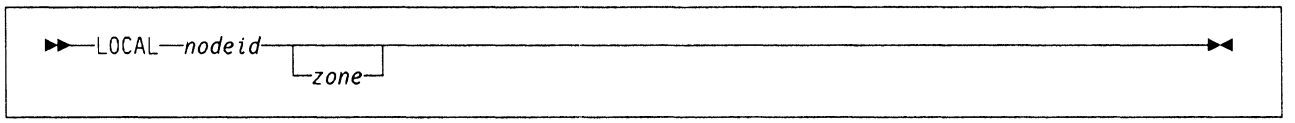
Operands

nodeid
is the name of the default selection node.

* indicates that the local node is the default selection node.

AUTO
specifies that the Initial Menu is bypassed and a session is initiated with the default selection node.

LOCAL



Purpose

Use the LOCAL record to define the name given to the local PVM system. The node identifier is also treated as an APPC/VM resource ID if PVM is to be recognized for APPC/VM communication and a RESID record does not appear in the PVM configuration file.

Usually, this record must be the first record in the configuration file. However, it may be preceded by the comment, LANG, and XDBC records.

Operands

nodeid
is the name of the local node. The *nodeid* can be 1 to 8 characters. *DEFAULT* is a reserved word and should not be used as *nodeid*.

zone
is the time zone from GMT. The *zone* operand is entered as an unsigned number (*nn*) if the zone is west of GMT, and as *-nn* if the zone is east of GMT.

Note: If DIAGNOSE code X'0' returns the time zone offset, then that value overrides *zone*.

MAXCONN

Purpose

Use the MAXCONN record to indicate the maximum number of IUCV and APPC/VM paths supported by PVM. Use this value to generate a table for storing path information. The value ranges from 100 to 4096. If this record is missing, the default value is 1096.

All LOCAL, LINK, PORT, CLUSTER, CLPORT, GROUP, ROUTE, or AUTHORIZ records must precede this record.

Operands

nnnn
indicates the maximum number of IUCV paths supported by PVM. This number should not exceed the MAXCONN value in the CP OPTION Directory record for PVM. The value of *nnnn* ranges from 100 to 4096.

MAXIDLE

Purpose

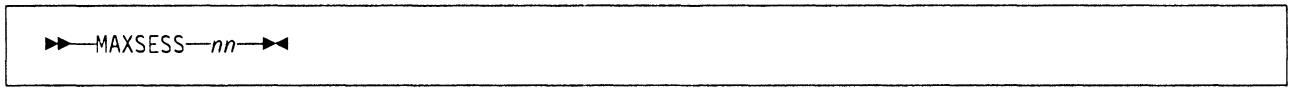
Use the MAXIDLE record to specify the maximum time a MPVM user may remain idle before PVM will terminate all of the user's sessions. The user is considered idle when there is no communication from the MPVM user to PVM for any session.

All LOCAL, LINK, PORT, CLUSTER, CLPORT, GROUP, ROUTE, or AUTHORIZ records must precede this record. If omitted, the default value is 600 seconds (10 minutes). If more than one MAXIDLE record is entered in a configuration file, only the last MAXIDLE record in the list is accepted, the others are ignored. PVM will send out a warning message when approximately three-fourths of the time interval has elapsed.

Operands

nnnn
Is the number of seconds allowed for an idle user. The value of nnnn ranges from 60 to 9998. A setting of 9999 will mean no maximum idle time limit.

MAXSESS



Purpose

Use the MAXSESS record to specify the maximum number of sessions a MPVM user may establish.

All LOCAL, LINK, PORT, CLUSTER, CLPORT, GROUP, ROUTE, or AUTHORIZ records must precede this record. If omitted, the default value is 5 sessions. If more than one MAXSESS record is entered in a configuration file, only the last MAXSESS record in the list is accepted, the others are ignored.

Operands

nn Is the number of sessions allowed for each MPVM user. The value of nn ranges from 1 to 99.

MAXSUSP

Purpose

Use the MAXSUSP record to specify the maximum time a MPVM user may have a session suspended before that user's session will be terminated by PVM. A session is a suspended if:

- It is not the current session
- The MPVM SUSPEND command was issued for the current session
- A CP or CMS command was issued to the host session during a current session.

All LOCAL, LINK, PORT, CLUSTER, CLPORT, GROUP, ROUTE, or AUTHORIZ records must precede this record. If omitted, the default value is 1200 seconds (20 minutes). If more than one MAXSUSP record is entered in a configuration file, only the last MAXSUSP record in the list is accepted, the others are ignored. PVM will send out a warning message when approximately one-half of the time interval has elapsed, and then again when approximately three-fourths of the time interval has elapsed.

Operands

nnnn
Is the number of seconds allowed for a suspended MPVM session. The value of nnnn ranges from 60 to 9998. A setting of 9999 will mean that there is no maximum suspend time limit.

MAXTDISC

Purpose

Use the MAXTDISC record to specify the maximum time a MPVM user may remain temporarily disconnected before all of the user's session will be terminated by PVM.

All LOCAL, LINK, PORT, CLUSTER, CLPORT, GROUP, ROUTE, or AUTHORIZ records must precede this record. If omitted, the default value is 1200 seconds (20 minutes). If more than one MAXTDISC record is entered in a configuration file, only the last MAXTDISC record in the list is accepted, the others are ignored. PVM will send out a warning message when approximately three-fourths of the time interval has elapsed.

Operands

nnnn
Is the number of seconds allowed for a temporarily disconnected MPVM user. The value of nnnn ranges from 60 to 9998. A setting of 9999 will mean no maximum temporary disconnect time limit. A setting of 0 will indicate that temporary disconnect for MPVM users is not allowed.

MAXUSERS

Purpose

Use the MAXUSERS record to specify the maximum number of MPVM users who may connect to PVM.

All LOCAL, LINK, PORT, CLUSTER, CLPORT, GROUP, ROUTE, or AUTHORIZ records must precede this record. If omitted, the default value is 100 MPVM users. If more than one MAXUSERS record is entered in a configuration file, only the last MAXUSERS record in the list is accepted, the others are ignored. The maximum number of users can also be limited indirectly through the number of IUCV and APPC/VM paths supported by PVM. This value can be changed through the MAXCONN record in the configuration file but cannot exceed the limit specified in the CP OPTION directory record for the PVM virtual machine.

Operands

nnn
Is the maximum number of MPVM users allowed. The value of nnn ranges from 1 to 999.

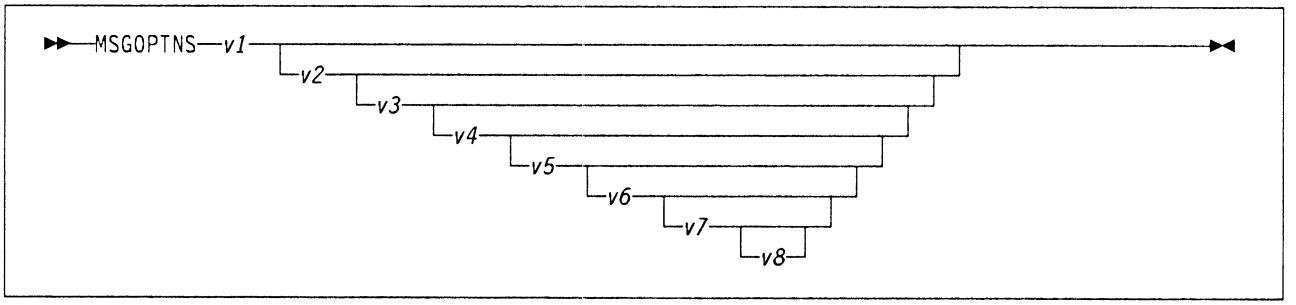
MSGNOH

Purpose

Use the MSGNOH record to indicate that responses to all commands entered by the CP SMSG command must be returned using the CP MSGNOH command rather than the CP MSG command, unless prohibited by the MSGOPTNS on that record when sending messages to the CP OPERATOR. The PVM user ID must have Class B privilege to issue the CP MSGNOH command.

All LOCAL, LINK, PORT, CLUSTER, CLPORT, GROUP, ROUTE, or AUTHORIZ records must precede this record. If you enter more than one MSGNOH record in the configuration file, only the last MSGNOH record in the list is accepted. PVM ignores the other MSGNOH records.

MSGOPTNS



Purpose

Use the MSGOPTNS record to indicate what options PVM will use regarding time stamps and message numbers for messages directed to the PVM user ID, to the audit file, to users, and to the operator. The installation can chose for each destination whether time stamps and message numbers should be appended to the messages. All parameters must be either a Y for yes or a N for no.

All LOCAL, LINK, PORT, CLUSTER, CLPORT, GROUP, ROUTE, or AUTHORIZ records must precede this record.

Operands

v1 specifies the time stamps on the messages to the PVM user ID. The default is N.

v2 specifies the message numbers on the messages to the PVM user ID. The default is N.

v3 specifies the time stamps on the messages to the PVM audit file. The default is Y.

v4 specifies the message numbers on the messages to the PVM audit file. The default is N.

v5 specifies the time stamps on the messages to users. The default is N.

v6 specifies the message numbers on the messages to users. The default is N.

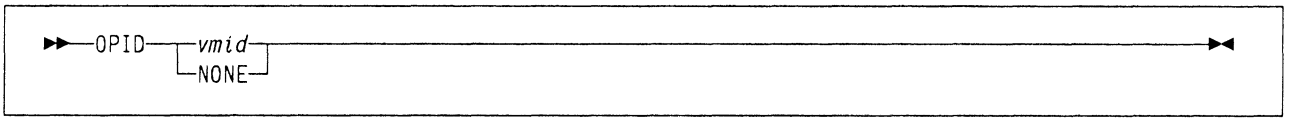
v7 specifies all messages written to the operator. These messages are written by a CP MSG command, overriding the MSGNOH record. The default is N, meaning that the MSGNOH option will take effect.

v8 specifies the message numbers on the messages to the operator. The default is N.

Usage Notes

1. The values for the PVM audit file affect the record length of that file. The length of the AUDIT file depends on whether the audit records contain message numbers and/or time stamps. The length of the AUDIT file is:
   - 129 bytes if the audit record contains neither time stamps nor message numbers
   - 140 bytes if the audit record does not contain time stamps but does contain message numbers
   - 129 bytes + *maximum timestamp length* + 1, if the audit record contains a time stamp but does not contain message numbers
   - 140 bytes + *maximum timestamp length* + 1, if the audit record contains a time stamp and message numbers.
2. The + *1* allows for a single blank following the timestamp.
3. *Maximum timestamp length* refers to the longer of the AM or the PM timestamp for the default language. However, if the default language is a DBCS language, the *maximum timestamp length* is the longest of the:
   - AM and PM timestamp for the default language
   - AM and PM timestamp for the backup language.

OPID



Purpose

Use the OPID record to specify the virtual machine ID that receives messages that usually go to the system operator. You can also use this record to stop messages from being sent to the system operator.

All LOCAL, LINK, PORT, CLUSTER, CLPORT, GROUP, ROUTE, or AUTHORIZ records must precede this record. If you omit this record, the default is OP.

Operands

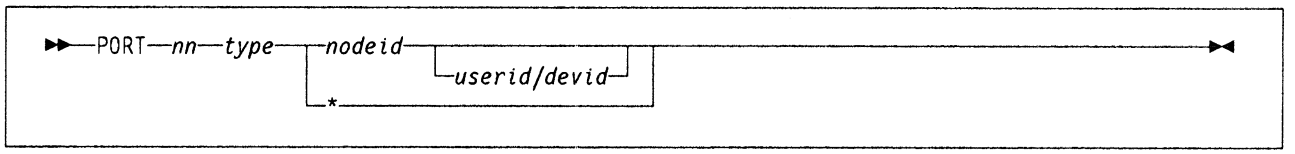
vmid
is the virtual machine to receive operator messages.

NONE
suppresses the messages sent to the operator.

Usage Notes

1. If you omit this record, operator messages are sent to the OPERATOR user ID.
2. The format of the messages sent to the specified operator depends on the options specified on the MSGOPTNS record and on whether PVM is authorized to use the CP MSGNOH command.
3. Only one *vmid* can be specified on this record.

PORT



Purpose

Use the PORT record to indicate that a port on the 327x line, defined in the prior LINK record, is accessible. You can also use the PORT record to specify the type of display station to be emulated.

All PORT records associated with a particular LINK record must immediately follow that record. PORT records for a given link need not be in numeric port number sequence.

Port numbers and types defined in the configuration file must be consistent with the definition of these ports at the remote node.

See "Configuration File PORT Record" on page 31 for information about using the PORT record to assure session security.

Operands

nn is the port number. It is a 1 or 2 digit number ranging from 0 to 31.

type

3277
is a 3277 Model 2 Display Station. Ports defined for a 3277 are also accessible by 3278 display stations. During session initiation, PVM looks for the first port that matches the display station exactly. If a 3278-defined port is not found for a 3278 type display station, PVM assigns an available defined 3277 port.

3278-2
is a 3278 Model 2 Display Station.

3278-3
is a 3278 Model 3 Display Station.

3278-4
is a 3278 Model 4 Display Station.

3278-5
is a 3278 Model 5 Display Station.

3279-2
is a 3279 Model 2 Display Station.

3279-3
is a 3279 Model 3 Display Station.

3284
is a 3284 Printer.

3286
is a 3286 Printer.

3287
is a 3287 Printer.

3288
is a 3288 Printer.

3289
is a 3289 Printer.

328X
indicates that the port is for any 3270 Information Display System printer. The device types supported are those listed above: 3284, 3286, 3287, 3288, 3289, and the 3262 model 3 and 13, when defined as a 3289.

nodeid
is the PVM node identifier of the users given exclusive use of this port. Refer to important related information in "Configuration File PORT Record" on page 31.

userid/devid
is the user or device given exclusive use of the port. This entry can be a 1-to-8-character user ID (*userid*) or a real device (display station or printer) address (*devid*).

Device addresses have the form:

nnn **or** *nnnn* for a local display station or printer.

nnnnn for a remote 3270 display station or printer that accesses PVM by the CP DIAL, PASSTHRU, or the CP NET ATTACH command.

CL*xx***-P***yy* for remote devices that access PVM through the remote 3270 line driver support.

For local devices, *nnn* (or *nnnn*) is the real address of the device. For remote devices, *nnnnn* is the resource identifier for that device. For remote devices driven by the PVM R3270 line driver, *xx* is the cluster number and *yy*

is the port number (on a node ID that is defined by a LINK record as type R3270).

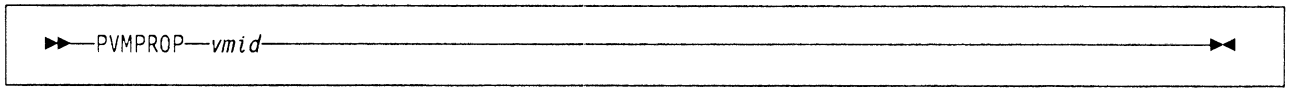
SNA devices cannot have ports reserved by device address or LU name because they have no corresponding real address. When a SNA device logs onto CMS, the resulting real address is null. To reserve a port in these cases, specify the associated VM user ID.

When reserving ports using **both** *userid* and *devid*, be aware that PVM assigns the lowest number available port with **either** a matching user ID or device ID. For important related information, see "Configuration File PORT Record" on page 31.

* specifies that no access restrictions apply to this port.

Note: The printer required by a target application must match (characteristics) the actual printer device handling its output. To ensure this match, it is recommended that the installation reserve specific ports for exclusive use by specific printers.

PVMPROP



Purpose

Use the PVMPROP record to identify a virtual machine running a programmed operator that receives a copy of PVM error messages. All messages are transmitted to the virtual machine by the CP Special Message (CP SMSG) Facility. The messages are sent to the PVMPROP virtual machine in the same encoded format as sent to the Command Response Interface. See Chapter 15, "Facilities for Automating PVM and PVMG Operations" on page 251 for more information about PVMPROP.

All LOCAL, LINK, PORT, CLUSTER, CLPORT, GROUP, ROUTE, or AUTHORIZ records must precede this record.

Operands

vmid
is the virtual machine to receive error messages. All PVM messages are sent to the *vmid* in encoded form using the CP SMSG command.

RESID

Purpose

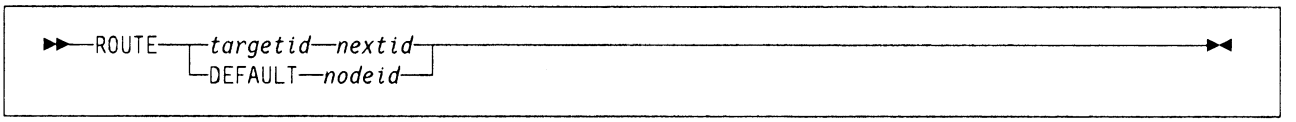
Use the RESID record to define a unique resource identifier. If the RESID record is not specified, the resource identifier defaults to the local node identifier specified in the LOCAL record.

The LOCAL record must precede the RESID record.

Operands

resourceid
is the APPC/VM resource identifier for this node. The *resourceid* must be unique within the system(s) accessing PVM using this name.

ROUTE



Purpose

Use the ROUTE record to define the next nodal path to a specified target node from the local PVM node. For PVM to PVM communications, routing must be consistently defined on both ends of network paths.

Use the **DEFAULT** *nodeid* form of the ROUTE record to define default routing for nodes not defined by LINK or ROUTE records. Only one default route record can be entered.

All LOCAL, LINK, PORT, CLUSTER, CLPORT, and GROUP records must precede this record.

Operands

targetid
is the identification of the target system node.

nextid
is the identification of the next PVM node defined in a configuration file LINK or GROUP record, in the routing pattern from the local node to the target system node.

DEFAULT *nodeid*
indicates that the *nodeid* is the default node identifier. The *nodeid* is the PVM node through which all unknown nodes are to be routed. This node must be defined in a LINK or GROUP configuration file record. To avoid difficulties, you should not define any PVM node identifiers with the name *DEFAULT*.

Note: Links that are members of groups should not be specified on ROUTE records.

SELECT

Purpose

Use the SELECT record to assign a frequently used node name to a PF key. You can specify up to 11 SELECT records to assign 11 PF keys The first SELECT record *nodeid* is assigned to PF1, the second SELECT record *nodeid* is assigned to PF2, and so forth to PF7. The eighth SELECT record is assigned to PF9 (PF8 is reserved for the SCROLL FORWARD function). The ninth SELECT record is assigned to PF10, and so forth to PF12.

If you use a SELECT record without specifying the node ID, the PF key associated with this SELECT record is omitted. This allows a PF key to be skipped.

All LOCAL, LINK, PORT, CLUSTER, CLPORT, GROUP, ROUTE, or AUTHORIZ records must precede this record.

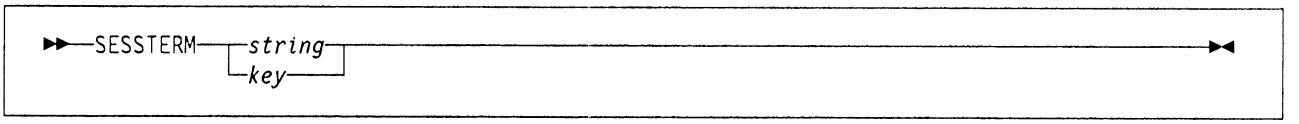
Operands

nodeid
is the identification of the node assigned to the PF key represented by this SELECT record. The *nodeid* may be 1 to 8 characters long. The name must not be a link name that is part of a GROUP definition.

Usage Notes

1. Two SELECT records in the configuration file cannot contain the same node name.

SESSTERM



Purpose

Use the SESSTERM record to define the default session termination string, PF*n*, or PA key used at the local node.

All LOCAL, LINK, PORT, CLUSTER, CLPORT, GROUP, ROUTE, OR AUTHORIZ records must precede this record. If this record is not entered in the configuration file, PVM uses the default string '####'.

Operands

string
is a 1-to-8-character session termination string. Mixed case strings are allowed.

key
is the key you wish to assign for a session termination key. Use the form *PFnn* or *PAn* to specify the key. For PF keys, *nn* can be a number from 01 (or 1) to 24. For PA keys, *n* can be 1 or 2.

STATTIME

Purpose

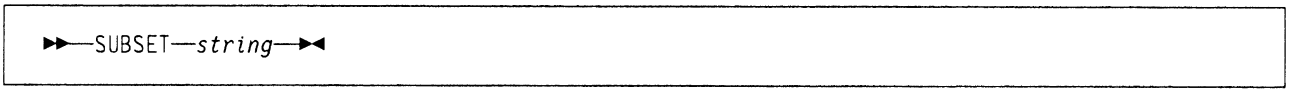
Use the STATTIME record to specify the time interval between calls to the Link Statistics Exit (DVMNMEL) for connected links. This is also the interval that timing records are sent down the network links for determining link delay time.

All LOCAL, LINK, PORT, CLUSTER, CLPORT, GROUP, ROUTE, or AUTHORIZ records must precede this record. If you omit this record, the default value is 600 seconds (10 minutes). If you enter more than one STATTIME record in the configuration file, only the last STATTIME record in the list is accepted. PVM ignores the other STATTIME records.

Operands

nnnn
is the number of seconds between calls to the Link Statistics Exit. The value of *nnnn* ranges from 10 to 1800.

SUBSET



Purpose

Use the SUBSET record to identify the command names that will be executed as CMS Subset Mode commands. Commands entered using this option do not produce the return code response and other responses, and they are not invoked with the *CMS* prefix.

All LOCAL, LINK, PORT, CLUSTER, CLPORT, GROUP, ROUTE, or AUTHORIZ records must precede this record, and all SUBSET records must appear together.

Operands

string
is a 1-to-8-character command name.

TDISC

Purpose

Use the TDISC record to specify the maximum length of time a PASSTHRU user may remain in a temporary disconnected state from a PVM session before PVM terminates the session.

All LOCAL, LINK, PORT, CLUSTER, CLPORT, GROUP, ROUTE, or AUTHORIZ records must precede this record. If you omit this record, the default value is 1200 seconds (20 minutes). If you enter more than one TDISC record in the configuration file, only the last TDISC record in the list is accepted. PVM ignores the other TDISC records.

Operands

nnnn
is the number of seconds to be allowed for temporary disconnect. The value of *nnnn* ranges from 30 to 9999.

TIMEOUT

Purpose

Use the TIMEOUT record to specify the time interval between line status checks by emulator line drivers and network line drivers when no data is being transferred. For an emulator line driver, if a POLL has not been received on the TP line within the specified time interval, the line driver terminates with an indication that the line is down. For the network line drivers, if there has been no activity on the line during the specified time interval, a transaction is sent down the line to ensure that the line is still up.

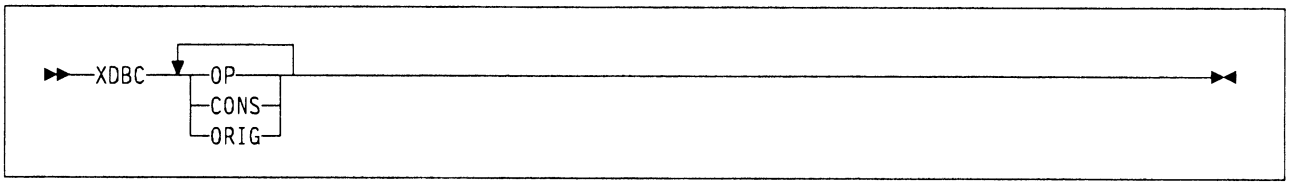
This record does not apply to R3270 type links, because the system continuously polls clusters on a remote line. This record does not apply to ROCF or IUCV line drivers. The ROCF line drivers use a self-defined interval of one minute.

All LOCAL, LINK, PORT, CLUSTER, CLPORT, GROUP, ROUTE, or AUTHORIZ records must precede this record. If you omit this record, the default value is 300 seconds (five minutes). If you enter more than one TIMEOUT record in the configuration file, only the last TIMEOUT record in the list is accepted. PVM ignores the other TIMEOUT records.

Operands

nnnn
is the number of seconds between status checks. The value of *nnnn* ranges from 30 to 9999.

XDBC



Purpose

Use the XDBC record to exclude certain users on the local system from receiving, in the national language, any PVM messages that, in the national language, contain double-byte characters. The identified users receive those messages in the backup language instead. The record may appear anywhere in the file.

The principal reason for the XDBC record is that PVM cannot tell whether the virtual machine consoles of other virtual machines on a VM system are capable of handling double-byte characters.

Operands

OP
identifies the CP Operator.

CONS
identifies the PVM user ID console (display station) and the AUDIT file.

ORIG
identifies all virtual machine users on the local VM system who are not PVM users, other than the CP operator and the PVM operator. That is, the originators of actions or commands to which PVM messages are sent in response.

The position of the XDBC record in the configuration file does not matter, and it takes effect immediately. If the LANG record identifies a DBCS language and the PVM user ID or the CP operator should not receive DBCS messages, the XDBC record should be specified immediately before or after the LANG record.

If PVM finds an error in the XDBC record, PVM immediately excludes OP, CONS, and ORIG from receiving DBCS messages. You must enter the CDBC command to enable messages to these destinations.

Appendix E. PVMG Commands

The PVMG administrator can use a variety of commands on the PVMG virtual machine to display system status, change system parameters, send commands to other PVMG nodes, or send messages to interactive session users.

These commands, when issued from PVMG's console, must be issued in the format:

pvmg *command operands...*

PVMG
is to signify a PVMG command follows.

Note: Throughout this chapter, we assume that the PVMG program is running in a virtual machine whose user ID is *PVMG*.

The pvmg prefix is assumed for all commands in this chapter.

Notes on Using Commands

Certain PVMG commands can be issued by users identified in AUTHORIZ records in the PVMG configuration file. These commands are the *A* commands listed in Table 30 on page 378. All PVMG commands are listed in this table. Some commands must be used by a PVMG operator logged onto the PVMG user ID. These commands are the *O* commands. A virtual machine user may use any of the unrestricted commands. These commands are the *G* commands. Some commands are intended to be used by the PVMG system programmer. These commands are the *S* commands.

Some PVMG command options have restriction levels that are different from the remaining options for that command.

If you are not working at the PVMG user ID, each PVMG command expression must be included in a CP SMSG (Special Message) command that contains the PVMG user ID. This sends the command to PVMG to be executed. The exception to this rule is when a local installation has provided an exec for each command that automatically puts the *SMSG pvmid* characters in front of the PVMG command expression.

You can use a command prefix when issuing PVMG commands. Command prefixes tell PVMG to send command responses using a particular format. See "Using Command Prefixes" on page 255 for more information about command prefixes. For information on how to use command prefixes in application programming, see Chapter 15, "Facilities for Automating PVM and PVMG Operations" on page 251.

Note: In applying prefixes the user should understand that the command *PVMG* is not a PVMG component command. It is a GCS command required to direct the command entered on the same line to the PVMG component. Therefore, prefixes should be applied to the particular command being entered, for example, Query.

© Copyright IBM Corp. 1991,1993

How to Read the Syntax Diagrams

Throughout this book, syntax is described using the structure defined below.

- Read the syntax diagrams from left to right, from top to bottom, following the path of the line.

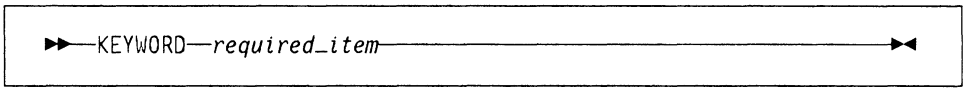
  The ▶▶— symbol indicates the beginning of a command.

  The —▶ symbol indicates that the command syntax is continued on the next line.

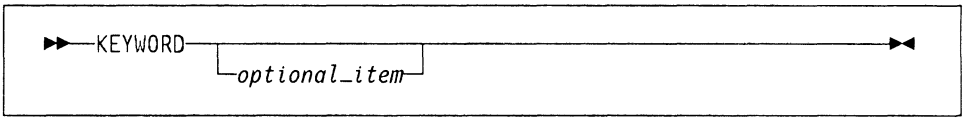
  The ▶— symbol indicates that a command is continued from the previous line.

  The —▶◀ symbol indicates the end of a command.

- Required items appear on the horizontal line (the main path).

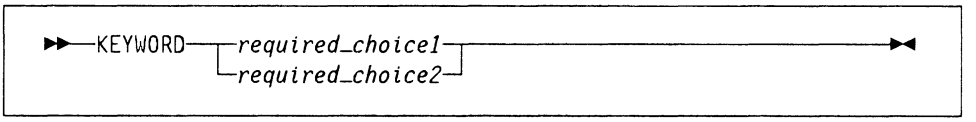


- Optional items appear below the main path.

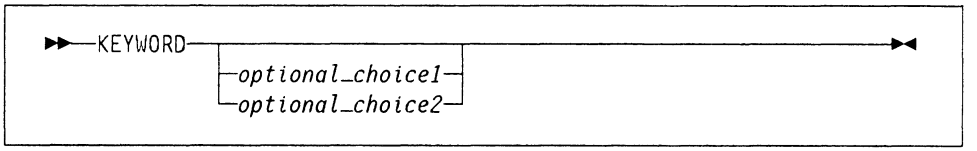


- If you can choose from two or more items, they appear vertically, in a stack.

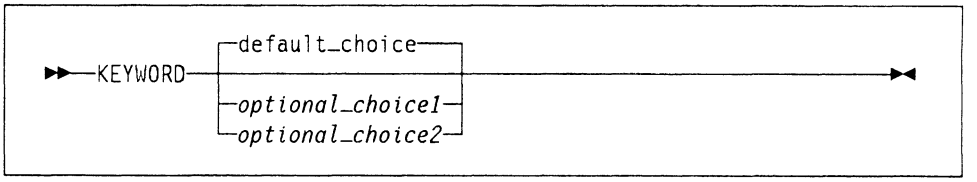
  If you *must* choose one of the items, one item of the stack appears on the main path.



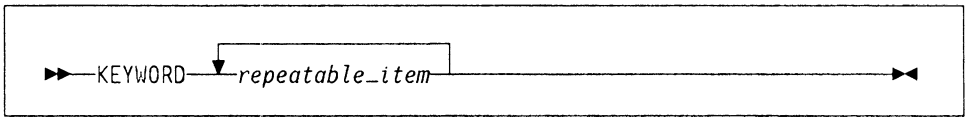
  If choosing one of the items is optional, the entire stack appears below the main path.



  If one of the choices is a default, it appears above the main path.

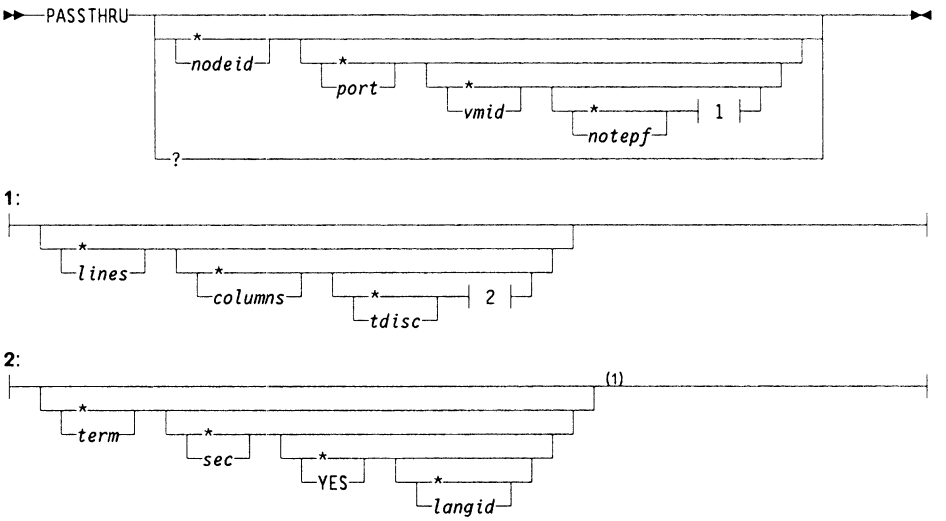


- An arrow returning to the left above the main line indicates an item that can be repeated.



A repeat arrow above a stack indicates that you can repeat the items in the stack.

The following example of the PASSTHRU command illustrates many aspects of a syntax diagram. Explanation of the PASSTHRU command is provided within the syntax diagram.



Note:

1 PASSTHRU is a positional command. If you wish to enter an option that follows other options that you do not wish to use, you must enter an asterisk (*) for each unused option that occurs before the option you wish to use. For example, if you wish to enter PASSTHRU with the *langid* operand, you must enter an asterisk as a placeholder for each preceding option. Your command would be PASSTHRU * * * * * * * * * *langid*. The ? operand cannot be followed by any other operand, since PASSTHRU ? provides help information about the PASSTHRU command.

Fragments *1* and *2* refer you to a continuation of the positional, sequential, operands. When you have completed entering operands, you can follow the syntax diagram to the end of the command, bypassing any remaining optional operands.

Table 30 lists the commands, their functions, and the users authorized to use them. Note the following about the first column of this table:

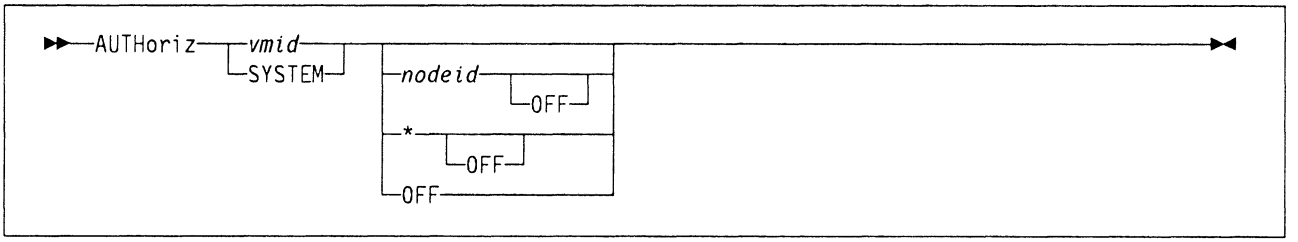
Table 29. Key to authorized users

User	Meaning
A	Alternate operator specified on an AUTHORIZ record in the PVMG configuration file.
O	PVMG operator logged onto the PVMG user ID.
G	General VM user.
S	PVMG system programmer.

Table 30. PVMG Commands and Execs and Their Functions

User	Command	Function
A O	AUTHORIZ	Adds or deletes entries from the PVMG authorized user list. This command must be issued on the local system.
A O	CDBC	Specifies whether display unit can or cannot handle messages in the default language without regard to double-byte character restrictions.
A O G	CMD	Sends PVMG commands to another PVM or PVMG node for execution.
O	DISC	Disconnects the PVMG user ID.
A O	DROP	Terminates an IUCV link to a PVM virtual machine, or a user session.
O	INIT	Starts PVMG initialization.
A O	LMSG	Sends a message to PVMG users on a specified link.
A O	LOGMSG	Defines the LOGMSG area on the PVMG Selection Menu.
A O G	MSG	Causes PVMG to forward a message to a specific VM user.
A O G	PMSG	Causes PVMG to forward a message to specified PVMG users. The ALL option is restricted to the *A* and *O* user groups.
S	PVMGBLD	Regenerates the PVMG LOADLIB.
A O G	QUERY	Displays information about PVMG. The AUTHORIZ option is restricted to the *A* and *O* user groups and to the local VM system.
A O	QUIESCE	Quiesces the PVMG server, or a line driver task.
A O	RESUME	Reverses the effect of the QUIESCE command.
A O	ROUTE	Adds, deletes, or changes a node's routing.
O	RUNPVMG	Loads and starts PVMG.
A O	SELECT	Adds, deletes, or changes nodes assigned to PF keys on the PVMG Selection Menu.
A O G	SHOW	Shows the path of a particular session.
O	SHUTDOWN	Terminates the PVMG server.
A O	SPOOL	Logs console messages and information to a virtual printer.
A O	START	Starts an IUCV link.
A O	TRACE	Starts or stops line tracing on a link, or system tracing on the PVMG server.

AUTHORIZ



Purpose

Use the AUTHORIZ command to add or delete entries from the PVMG authorized user list. To make this change permanent, the installation must add or delete an AUTHORIZ record in the PVMG configuration file.

Operands

vmid
is the user identification of the CP virtual machine you want to add to or delete from the authorization list. Without specifying * or *nodeid*, this authorization is for the local node only.

SYSTEM
authorizes the PVMG operator to issue commands to nodes (* or *nodeid*) other than the local node. Use this operand rather than the PVMG *vmid*. Without specifying * or *nodeid*, this authorization is for the local node only.

nodeid
is the specific node where the *vmid* is authorized. You can specify the same *vmid* from different *nodeid*s.

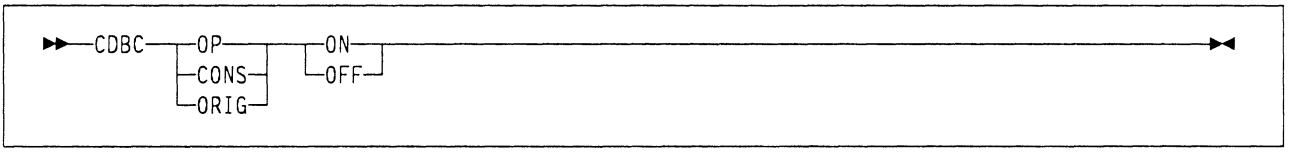
* indicates that *vmid* is authorized on any node. Consider this option carefully, since it may be a security risk in a network where user IDs could be duplicated. You cannot add *vmid* to the authorization list using both a global node ID (*) and a specific node ID.

OFF
removes the specified user from the authorization list. Specifying OFF as the second operand will result in using the local *nodeid* as the default specified node. This parameter is not validated against the list of link or route names provided to PVMG by the LINK and ROUTE records.

Usage Notes

1. This command is restricted to the PVMG operator or an alternate operator specified on an AUTHORIZ record in the PVMG configuration file and must be issued on the local system.
2. If the operator makes a typographic error in issuing the AUTHORIZ command, PVMG may create an entry in its AUTHORIZ table using this incorrect information. If the operator recognizes this error, delete the entry using the **OFF** option and submit the correct AUTHORIZ command again.
3. Use the QUERY AUTHORIZ command to display the contents of the AUTHORIZ table.
4. Without specifying * or *nodeid*, this authorization is for the local node only.
5. When issued from another virtual machine using SMSG, you can use a command prefix with this command to tell PVMG how to respond to the command. See "Using Command Prefixes" on page 255 for more information about command prefixes.

CDBC



Purpose

Use the CDBC (Change Double-Byte-Characters) command to change the DBCS-exclusion settings for OP, CONS, and ORIG without having to add an XDBC record to the PVMG Configuration File and re-initialize PVMG. These settings are ignored when a user (other than the PVMG console operator), specifically requests a language by using a command prefix.

Operands

OP
identifies the CP operator.

CONS
identifies the PVMG console.

ORIG
identifies all virtual machine users on the local VM system, other than the CP operator and the PVMG operator. That is, ORIG identifies the originators of actions or commands to which PVMG messages are sent in response.

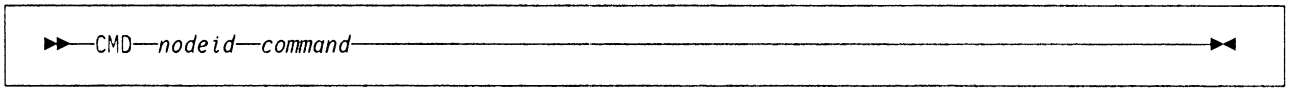
ON
indicates that the default language messages should be issued to the user without regard to DBCS content.

OFF
indicates that the backup language should be used for any messages to the user whenever the default language message that would otherwise be used contains DBCS.

Usage Notes

1. This command is restricted to the PVMG operator or an alternate operator specified on an AUTHORIZ record in the PVMG configuration file.
2. Issue the CDBC CONS OFF command when you are logged on to the PVMG user ID to prevent errors. You can use CDBC CONS ON when you are disconnected.
3. When issued from another virtual machine using SMSG, you can use a command prefix with this command to tell PVMG how to respond to the command. See "Using Command Prefixes" on page 255 for more information about command prefixes.

CMD



Purpose

The CMD command provides the user the ability to send a command to be executed at another PVM or PVMG node in the PVM network. You must be authorized at the remote node to have restricted commands issued at that node. The originator receives the output generated from this command. The CMD command is useful for various query commands.

Operands

nodeid
is the node on the PVM network where the command is to execute. This may be a PVM node or another PVMG node.

command
is any PVMG or PVM command you are authorized to issue.

Usage Notes

1. The CMD command lets you send commands from your PVMG server to a remote PVM or PVMG node, and then returns the results to you. For example, suppose you enter the command:

   ```
   pvmg cmd other q sys
   ```

   This sends a *QUERY SYSTEM* command from your PVMG virtual machine through the PVM network to the node called *OTHER*. The PVM or PVMG server at the node called OTHER issues the command and then sends the results back to you.

2. When issued from another virtual machine using SMSG, you can use a command prefix with this command to tell PVMG how to respond to the command. Place the prefix in front of the CMD command, rather than in front of the *command* portion of the command. For example, use the command:

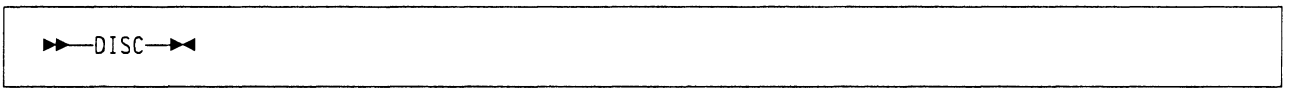
   ```
   smsg pvmg (1.kanji) cmd rmtpvmg5 query system
   ```

   instead of:

   ```
   smsg pvmg cmd rmtpvmg5 (1.kanji) query system
   ```

   See "Using Command Prefixes" on page 255 for more information about command prefixes.

DISC



Purpose

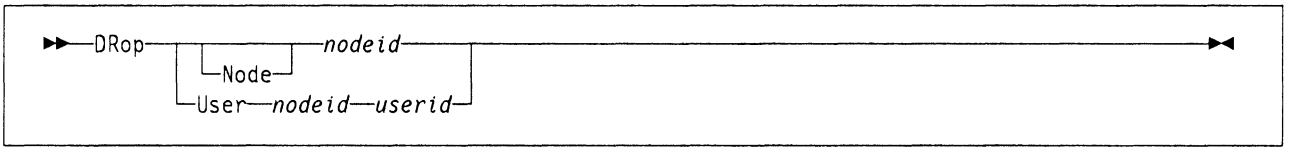
Use the DISC command to disconnect the PVMG operator display station. Use this command rather than the CP DISC command. Using the DISC command allows system spool files to be properly maintained.

Usage Notes

1. This command is restricted to the PVMG operator logged on to the PVMG user ID.
2. If the PVMG SPOOL CONSOLE START command has been issued, all "writes" or output messages, including those from PVMG, sent to your virtual console will continue to be captured.
3. If you log on PVMG, disconnect while SET RUN is OFF, then reconnect, you will not see any log messages or a RECONNECTED message. You will only see the CP read status. The messages will appear after you issue CP BEGIN.

   However, if you disconnect while SET RUN is ON, then RECONNECT, you will receive log messages and a RECONNECTED message.

DROP



Purpose

Use the DROP command to terminate a:

- Specific IUCV link to a PVM virtual machine
- User session.

Operands

Node *nodeid*
immediately terminates the specified PVMG link. Any active user sessions associated with this link are also terminated. The *nodeid* is the node name defined in the PVMG configuration file.

User *nodeid userid*
terminates the specified user session. The user is notified of impending system action. The *nodeid* is the identification of the user's origin or destination node where the user's session is terminated. The *userid* is the identification of the user whose PVMG session is terminated.

The session identification is:

- GRAF*xxx* for PVM dialed users
- the user's user ID for PASSTHRU users
- *SES*nnnn* for an MPVM session
- CL*cc*-P*pp* for a remote cluster user attached to a PVM node
- Real address of a display station that is locally attached to a PVM node,
- a logical device address if the session destination is a PVM node,
- a PVMG session ID (GRAF*xxxx*), if the session destination is an SNA application, or
- an LU name, if the session origin is an SNA application.

Usage Notes

1. This command is restricted to the PVMG operator or an alternate operator specified on an AUTHORIZ record in the PVMG configuration file.
2. When issued from another virtual machine using SMSG, you can use a command prefix with this command to tell PVMG how to respond to the command. See "Using Command Prefixes" on page 255 for more information about command prefixes.

INIT

Purpose

Use the INIT command to start PVMG initialization.

During initialization, PVMG processes its configuration file records, attaches its execution tasks, and opens its VTAM access control block (ACB). The INIT command cannot be executed again until PVMG has been shutdown.

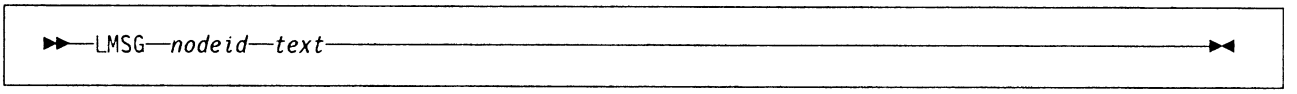
Operands

TRACE
TRACE indicates that system wrap trace is to be invoked during PVMG initialization. PVMG will generate a 16K wrap table for storing trace records. Tracing is stopped using the TRACE command.

Usage Notes

1. The INIT command is restricted to the PVMG operator.
2. The INIT command may be processed by the PROFILE GCS exec, which executes after the GCS recovery machine is IPLed, or it can be issued from the PVMG console.

LMSG



Purpose

Use the LMSG command to send a message to all PVMG users associated with a specific link, or to all interactive users who have sessions that access the PVMG server through VTAM, or that access VTAM through PVMG. If the name is the local PVMG node name, the message is sent to all users, except those with PVM-to-PVM sessions using the PVMG server as an intermediate node.

When the message arrives at the specified user's system, it replaces the current image on the user's display. The user recalls the stored image by pressing any interrupt key. The sender's node ID and VM ID are displayed with the message at the receiving location(s).

Operands

nodeid
specifies a specific VM node ID for users who have interactive sessions using PVMG, or the local PVMG node name.

text
is the text of the message sent. The message can be of any format and characters acceptable to the CP command processor. The total length is limited to the CP command input limit.

Usage Notes

1. This command is restricted to the PVMG operator or an alternate operator specified on an AUTHORIZ record in the PVMG configuration file.
2. No error message is issued if the specified link has no sessions.
3. When issued from another virtual machine using SMSG, you can use a command prefix with this command to tell PVMG how to respond to the command. See "Using Command Prefixes" on page 255 for more information about command prefixes.

LOGMSG

Purpose

The operator uses the LOGMSG command to enter the variable data text that appears in the LOGMSG area on the PVMG Selection Menu.

Any entry of the LOGMSG command overlays the previous LOGMSG text. Issuing a LOGMSG command with no text overlays the previous LOGMSG text with blanks.

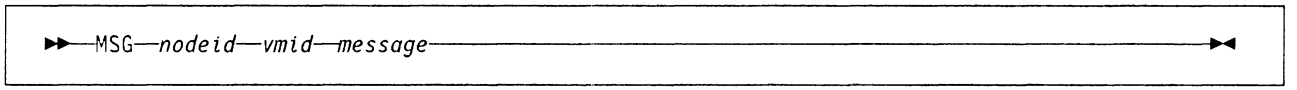
Operands

text
is the text of the message. You may use any characters. This field may be from 0 to 125 characters long, using two lines. If you use LOGMS, the message could only be up to 124 characters long. If you use LOGMSG, the message could only be up to 123 characters long.

Usage Notes

1. This command is restricted to the PVMG operator or an alternate operator specified on an AUTHORIZ record in the PVMG configuration file.
2. When issued from another virtual machine through SMSG, the text operand length may be up to 136 characters long.
3. When issued from another virtual machine through SMSG, you can use a command prefix with this command to tell PVMG how to respond to the command. See "Using Command Prefixes" on page 255 for more information about command prefixes.

MSG



Purpose

MSG causes a line of text to be sent to the specified node and presented to a user or operator using the CP MESSAGE command.

Operands

nodeid
is the identifier of the target node that is to receive the specified message text line. This node can be the local node.

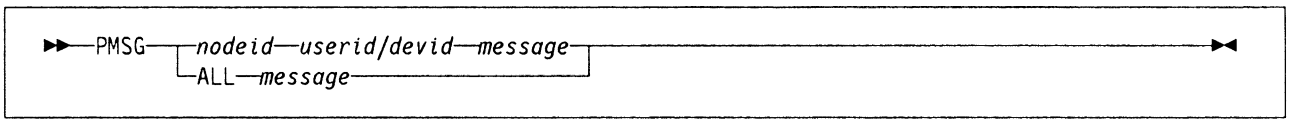
vmid
is the identifier of a logged on user at that location.

message
is the text of the message to be forwarded. The message can be of any format and characters acceptable to the CP command processor. The total message length is limited to the CP command input limit.

Usage Notes

1. When issued from another virtual machine using SMSG, you can use a command prefix with this command to tell PVMG how to respond to the command. See "Using Command Prefixes" on page 255 for more information about command prefixes.

PMSG



Purpose

Use the PMSG command to send a message to a particular user or all users who have sessions with or through PVMG. When the message arrives at the specified user's system, it replaces the current image on the user's display. The user recalls the stored image by pressing any interrupt key. The sender's node ID and VM ID are displayed with the message at the receiving locations.

Operands

nodeid
is the identification of the target user's node when the message is forwarded to a single user.

userid/devid
is the identification of the user (*userid*) or the identification of the user's attached display support task (*devid*) when the message is forwarded to a single user. This command is not valid to printer devices.

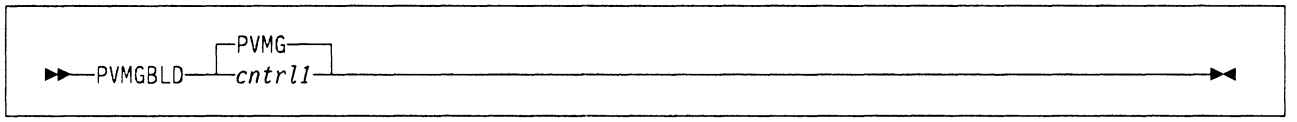
ALL
forwards the message to all PVMG users whose sessions originate, terminate, or pass through this PVMG server machine.

message
is the text of the message forwarded. The message can be of any format and characters acceptable to the CP command processor. The total length is limited to the CP command input limit.

Usage Notes

1. The ALL option of this command is restricted to the PVMG console operator or an alternate operator specified on an AUTHORIZ record in the PVMG configuration file.
2. When issued from another virtual machine using SMSG, you can use a command prefix with this command to tell PVMG how to respond to the command. See "Using Command Prefixes" on page 255 for more information about command prefixes.

PVMGBLD (PVM 2.1)



Purpose

Use the PVMGBLD command when regenerating the PVMG LOADLIB.

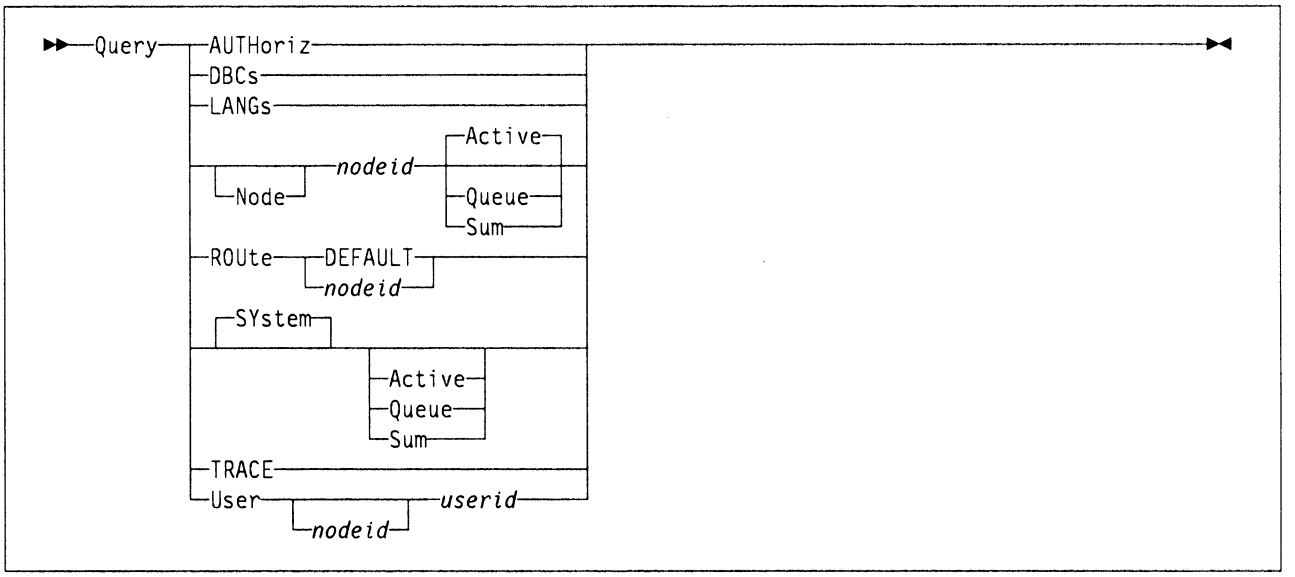
Operands

cntrl1
specifies the name of a LKEDCTRL file that contains the names of object modules in the order in which they are to reside in the resulting load module that is placed in the LOADLIB. The default value is **PVMG**. It creates a load library containing a single load module.

Usage Notes

1. This command is intended for use by the PVMG system programmer.
2. A filemode A minidisk must be accessed in write mode to successfully execute this command.
3. A PVMG LKEDIT (or *CNTRL1*) is also produced.

QUERY



Purpose

Use the QUERY command to display:

- Users authorized to issue restricted PVMG commands
- Contents of the routing table
- System status
- User session information
- Current DBCS settings
- National languages enabled on a PVMG system
- Current status of a given link
- Current trace settings.

If you enter the QUERY command with no operands, QUERY SYSTEM is the default.

Operands

AUTHoriz
displays the current list of authorized users in the system who can enter PVMG restricted commands.

DBCs
displays the DBCS settings for CP operator, the PVMG console, and the VM users when the default language for the PVMG virtual machine is a DBCS language.

LANGs
displays the language identifiers of the languages for which PVMG is enabled. LANGs also displays the languages used as the default national language and as the backup language at the local node.

Node *nodeid*
displays the status of the specified link.

Active
displays the status of the specified link, if active. Active is the default if Queue, or Sum are not specified.

Queue
displays the status of the specified link, and the associated user sessions for the specified link.

Sum
displays the statistics of the link, including the number of transaction counts (the number of data blocks)

ROUte
displays the contents of the PVMG server's routing table for either the DEFAULT route or a particular route name.

DEFAULT
displays the default route name.

nodeid
the node's entry from the routing table is displayed.

SYstem
displays information about the status of the system. If you do not specify any other operands, the status of each link defined within the system is displayed. QUERY SYstem is the default value if you enter the QUERY command with no operands.

Active
displays the status of each active link. If there are no active links, only the name of the local system is displayed.

Queue
displays the status and the associated user sessions for each active link.

Sum
displays the status of each link, including transaction counts

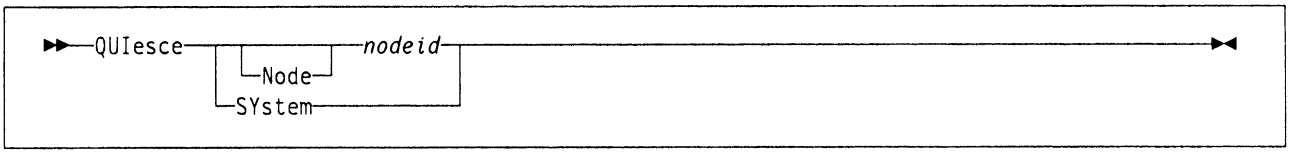
TRACE
displays trace status

User *nodeid userid*
displays information about the associated session identified by *nodeid* and *userid*. The specified session must be known to the PVMG server. The response includes the user's real display station address, type, and model number, along with the session origin and destination IDs.

Usage Notes

1. The AUTHorize operand is restricted and must be issued from a user ID on the local system.
2. When issued from another virtual machine using SMSG, you can use a command prefix with this command to tell PVMG how to respond to the command. See "Using Command Prefixes" on page 255 for more information about command prefixes.

QUIESCE



Purpose

The PVMG operator uses the QUIESCE command to quiesce:

- The PVMG server
- A line driver task.

Quiesce is a term used to describe the state of a task (or PVMG itself). A task in a quiesce state does not allow any new sessions and terminates when all active sessions have ended.

The effect of the QUIESCE command can be reversed by issuing the RESUME command. QUIESCE does not terminate any line drivers that have active sessions.

Operands

Node *nodeid*
indicates that the node *nodeid* is quiesced. After issuing this command, no new users are allowed to initiate sessions through the associated *nodeid*. When the last user session on the *nodeid* terminates, the node terminates. If no user session is active, the specified *nodeid* terminates immediately.

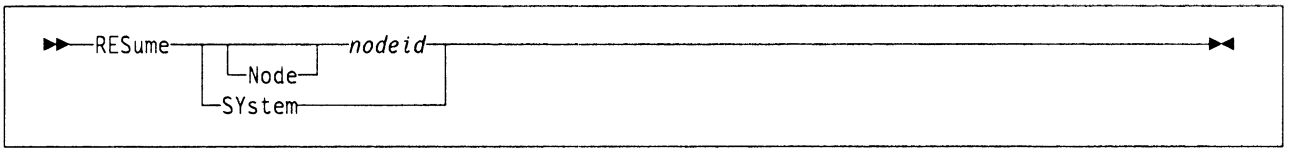
The *nodeid* must be the same name defined in the configuration file for the corresponding node.

SYstem
indicates that the PVMG server is quiesced. After issuing this command, no new users are allowed to initiate sessions through the PVMG server.

Usage Notes

1. This command is restricted to the PVMG operator or an alternate operator specified on an AUTHORIZ record in the PVMG configuration file.
2. When issued from another virtual machine using SMSG, you can use a command prefix with this command to tell PVMG how to respond to the command. See "Using Command Prefixes" on page 255 for more information about command prefixes.

RESUME



Purpose

Use the RESUME command to reverse the effect of a previously issued QUIESCE command. As a result, the specified line, or the entire system is made available.

Note: If the quiesced line driver has already terminated due to the QUIESCE command, use the START command instead of the RESUME command.

Operands

Node *nodeid*
indicates that the use of line defined by *nodeid* is resumed. Following the issuing of this command, new users are allowed to initiate sessions on the affected *nodeid*.

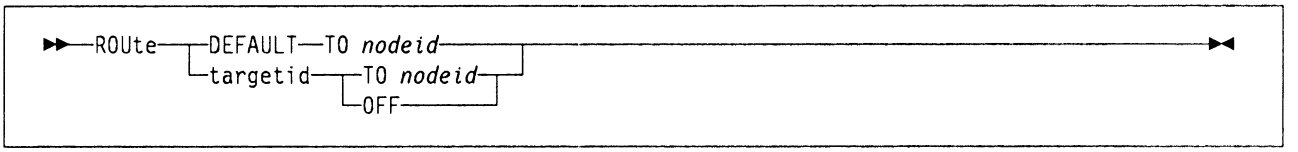
The *nodeid* must be the name defined in the configuration file for the corresponding line.

SYstem
indicates that the use of the PVMG server is resumed. After issuing this command, new users are allowed on the PVMG server.

Usage Notes

1. This command is restricted to the PVMG operator or an alternate operator specified on an AUTHORIZ record in the PVMG configuration file.
2. When issued from another virtual machine using SMSG, you can use a command prefix with this command to tell PVMG how to respond to the command. See "Using Command Prefixes" on page 255 for more information about command prefixes.

ROUTE



Purpose

The ROUTE command modifies the in-storage routing table used by PVMG for session routing. Node names may be added, deleted, or have their routing altered by the command.

Operands

DEFAULT
changes the DEFAULT routing.

targetid
is the target node added, deleted, or modified.

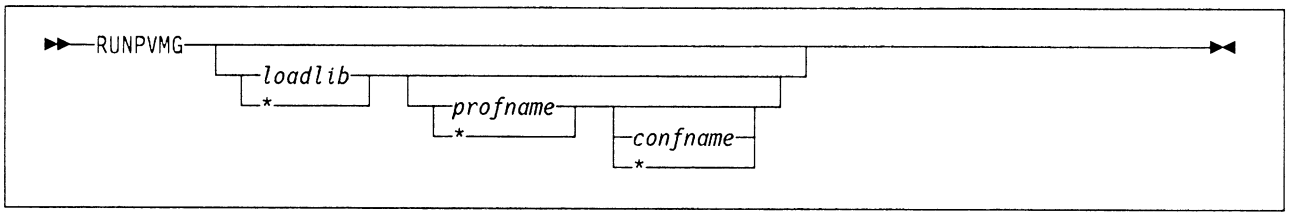
TO *nodeid*
indicates that the target node is routed through the link *nodeid*. The *nodeid* is the node name of the link. This node must be a PVM node directly connected to the local PVMG server.

OFF
removes the specified target node from the routing table. OFF may not be specified as a default route.

Usage Notes

1. This command is restricted to the PVMG operator or an alternate operator specified on an AUTHORIZ record in the PVMG configuration file.
2. When issued from another virtual machine using SMSG, you can use a command prefix with this command to tell PVMG how to respond to the command. See "Using Command Prefixes" on page 255 for more information about command prefixes.

RUNPVMG



Purpose

Use the RUNPVMG command to load and start PVMG.

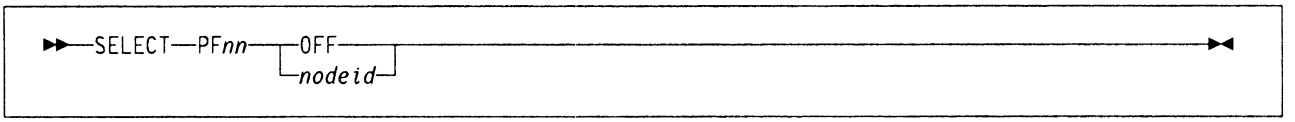
Operands

loadlib
is the name of the PVMG loadlib loaded. If you specify *, the default file name, *loadlib*, is PVMG. The file type is LOADLIB.

profname
is the name of the PVMG exec executed at startup time. If you specify *, the default file name, *profname*, is PVMG. The file type of this exec is GCS.

confname
is the name of the configuration file. If you specify *, the default file name, *confname*, is PVMG. The file type of this configuration file is CONFIG.

SELECT



Purpose

Use the SELECT command to dynamically add, delete, or modify the nodes assigned to PF keys by the SELECT records of the configuration file.

Operands

PF*nn*
is the PF key modified. The value of *nn* ranges from 1 to 12, excluding 8.

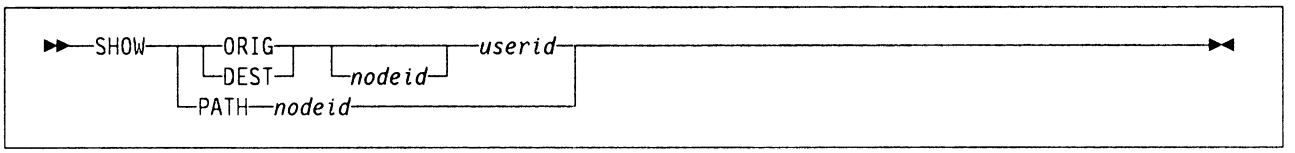
OFF
indicates that the PF key definition is to be reset

nodeid
is the name of the node selected by the specified PF key.

Usage Notes

1. This command is restricted to the PVMG operator or an alternate operator specified on an AUTHORIZ record in the PVMG configuration file.
2. When issued from another virtual machine using SMSG, you can use a command prefix with this command to tell PVMG how to respond to the command. See "Using Command Prefixes" on page 255 for more information about command prefixes.

SHOW



Purpose

Use the SHOW command to show the path of a particular session, or the path to a particular node. The system on which the command is first executed locates the session and displays information indicating that it is part of the path. This message gives the link upon which the session comes in, and the link upon which the session goes out.

Operands

ORIG
traces the path to the origin. If you use ORIG and the origin is not the local system or the incoming link, the command is forwarded to the node connected to the incoming link for processing.

DEST
traces the path to the destination. If you use DEST and the destination is not the local system or the outgoing link, the command is forwarded to the node connected to the outgoing link for processing. In this way, the command is executed on each intermediate node along the path.

PATH
shows the PVM routing to get to a particular target node.

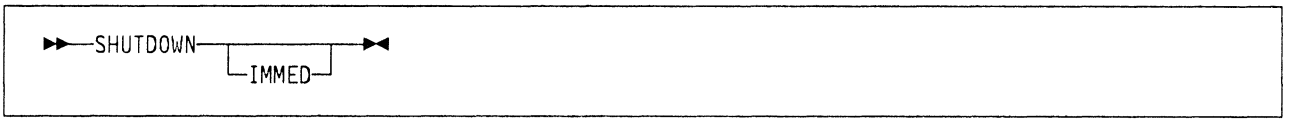
nodeid
is the node name of the origin or destination of the session. For ORIG or DEST, if you do not specify a value for *nodeid*; the local node name is assumed. For PATH, you must specify a value for *nodeid*, and it is the target node for which path information is required.

userid
is the PVMG identification for the origin or destination of the session.

Usage Notes

1. SHOW PATH shows the PVMG routing to get to a particular target node. The command locates the outbound path to the target node. If the target node is the local node or a nonnetwork node, you receive messages that tell you the path to the target node, and the message `COMMAND COMPLETE`. If the target node is not the local node or a nonnetwork node, you receive the message `Path to node xxxxx is through node yyyyy`. Then the SHOW command is forwarded to the node yyyyy for execution.
2. When issued from another virtual machine using SMSG, you can use a command prefix with this command to tell PVMG how to respond to the command. See "Using Command Prefixes" on page 255 for more information about command prefixes.

SHUTDOWN



Purpose

Use the SHUTDOWN command to conditionally terminate the PVMG server. If the PVMG server has no active sessions, then PVMG will shut down immediately. However, if the PVMG server determines that there are active sessions, it will not shut down until all active sessions end. All line drivers are placed in a quiesced state. Line drivers terminate when the last users of the associated lines terminate their sessions. PVMG terminates when the last user session terminates and all associated tasks terminate with it unless the IMMED

option is specified. The operator can then wait for the sessions to terminate, or issue the SHUTDOWN IMMED command to shut down immediately.

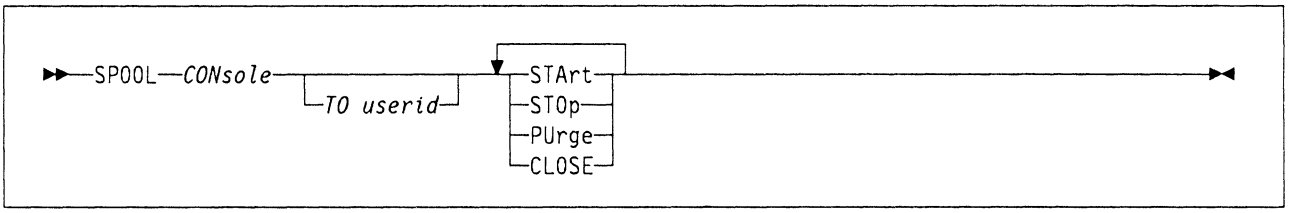
Operands

IMMED
terminates the PVMG server, canceling all active sessions.

Usage Notes

1. This command is restricted to a PVMG operator logged onto the PVMG user ID.
2. For SHUTDOWN IMMED, sessions are terminated immediately.

SPOOL (PVM 2.1.1)



Purpose

Use the PVMG SPOOL command to log console messages and information to a virtual printer to transfer later to a user ID on the same system. This command can be used for monitoring the PVMG machine.

Operands

CONsole
modifies the options for the virtual console spool file and/or initiates or stops the spooling of virtual console input and output, including CP input/output and PVMG console messages.

TO userid
sends the output of the virtual device to the virtual reader of the specified userid. The distribution code on the spool file retains the distribution code specified for the originator in the system directory.

CLOSE
closes the console spool file. If PURGE is specified, CLOSE is not necessary.

PUrge
closes and purges the console spool file. If PURGE is specified, CLOSE is not necessary. PURGE is equivalent to entering CLOSE with the PURGE option.

STArt
begins placing input and output from the PVMG console into a spool file. After the console is closed, the file becomes a printer spool file to be transferred to the userid, if specified.

STOp
terminates the spooling of the PVMG console. STOP does not close the console spool file.

Usage Notes

1. The START and STOP operands can not be specified together.
2. When disconnecting from the PVMG console, use the PVMG DISC command instead of the CP DISC command so PVMG will continue to issue messages to the console.

START

Purpose

Use the START command to start an IUCV link. The START command does not have to be invoked during PVMG initialization. PVMG attempts to connect each line defined with a configuration file LINK record during initialization. Use the START command if, after initialization, a link needs to be placed in the connect state.

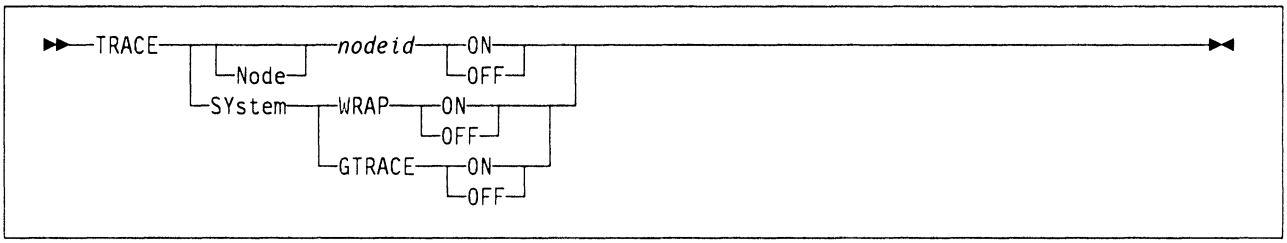
Operands

Node *nodeid*
is the node ID to be started. The *nodeid* must be defined by a configuration file LINK record.

Usage Notes

1. This command is restricted to the PVMG operator or an alternate authorized operator.
2. When issued from another virtual machine using SMSG, you can use a command prefix with this command to tell PVMG how to respond to the command. See "Using Command Prefixes" on page 255 for more information about command prefixes.
3. If the link on the PVM side is not active, PVMG will attempt the IUCV connect at 60 second intervals until either communications are established, or a PVMG DROP node command is issued by an operator.

TRACE



Purpose

Use the TRACE command to start or stop tracing on a link.

Operands

Node *nodeid*
specifies the node name (*nodeid*) associated with the PVMG link for which tracing is requested. The node name specified must be a valid node name as specified by a LINK record. Trace information is written to a wraparound buffer area for a particular link.

PVMG allocates an 8K byte buffer storage area for a wraparound trace table for each node for which tracing is started. Line driver activity is written as trace records to this table.

SYstem
causes a trace for the PVMG component. Trace information is written to a PVMG managed wraparound table. Specifying OFF terminates only the specified system tracing option. Other system trace options are not terminated.

WRAP
causes wraparound tracing for system events. Trace records are written to a 16K byte wraparound trace table managed by PVMG.

GTRACE
writes the output to a GCS trace table. These events may also be copies to a CPTRAP file on non VM/ESA or VM/XA systems. The trace records may be accessed later by the CP TRAP or the IPCSSCAN function.

ON
initiates the trace. If the GTRACE option is specified, the GCS ITRACE command is invoked.

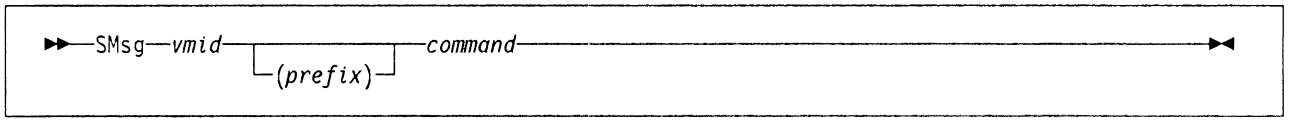
OFF
terminates the trace. If the GTRACE option is specified, the GCS ITRACE command is reset.

Usage Notes

1. This command is restricted to the PVMG operator or an alternate operator specified on an AUTHORIZ record in the PVMG configuration file.
2. When issued from another virtual machine using SMSG, you can use a command prefix with this command to tell PVMG how to respond to the command. See "Using Command Prefixes" on page 255 for more information about command prefixes.

CP Command Used with PVMG

SMSG



Purpose

Use the CP SMSG command to issue PVMG commands to PVMG from another virtual machine.

Operands

vmid
is the virtual machine identification of the PVMG user ID.

prefix
is any valid PVMG command prefix. See"Using Command Prefixes" on page 255 for more information about command prefixes.

command
is any valid PVMG command.

Appendix F. PVMG Configuration File Records

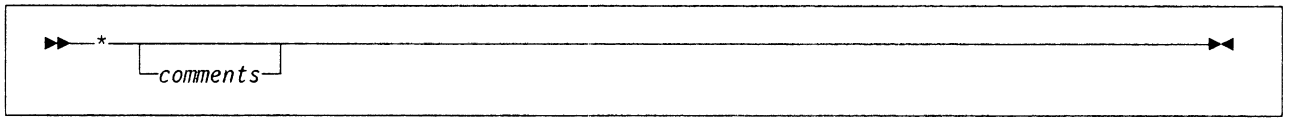
The PVMG configuration file (PVMG CONFIG) provides installation independent information to the PVMG service machine. It is a fixed length 80 byte file. Except for comments and *string values*, input must be in upper case, and is restricted to 80 columns with no continuation. Operands in a configuration file record must be separated by one or more blanks. During PVMG initialization this file is processed and used to define execution attributes. Use the * record to imbed comments in the configuration file.

PVMG configuration file records do not have to be sequenced in a particular order. PVMG process records using two passes. If errors are encountered, PVMG terminates following processing of the records.

This chapter includes a description of each record type.

---

*



Purpose

Use the * record to imbed comments in the PVMG configuration file. This record is ignored during configuration file processing. You can place this record anywhere in the PVMG configuration file.

Operands

comments
is any text message. This information must follow the asterisk(*).

Usage Notes

1. The Comment record must begin in column one of the PVMG configuration file.
2. There can be no blank lines in the PVMG configuration file. To insert lines between records, use this record.

© Copyright IBM Corp. 1991,1993

APPLN

Purpose

Use the APPLN record to define alternate names for VTAM applications accessible through PVMG. These alternate names can be specified by users requesting sessions with these applications that are defined in ROUTE table entries at other PVM nodes.

Operands

name
is the name associated with the VTAM application name provided on the second parameter.

applid
is the VTAM application name as it is known to VTAM

Usage Notes

1. All names defined in the PVMG configuration file must be known to the originating PVM node for session requests for the target *applid*. Names are made known to the originating PVM node by defining them in the route table (using the ROUTE record in the PVM configuration file) for that PVM server virtual machine. For example, a local configuration:

   - PVM node named PVM1
   - PVMG node named PVMG1
   - VTAM application named TSO1

   The installation wants to refer to TSO1 as OURTSO.

   The PVM configuration file will contain a ROUTE record:

   ```
   ROUTE OURTSO PVMG1
   ```

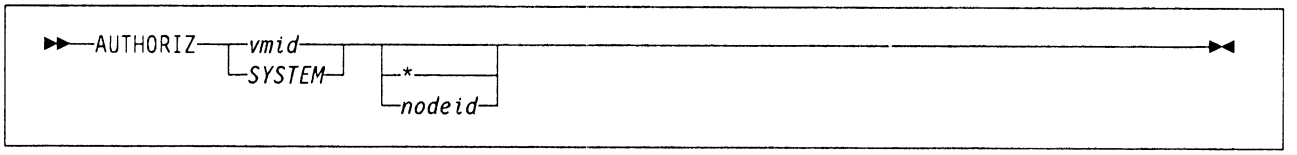
   The PVMG configuration file record must contain the APPLN record:

   ```
   APPLN OURTSO TSO1
   ```

   References to the name OURTSO by PVM users are translated to TSO1 application requests. Likewise, references to TSO1 by PVM users will also work if the PVM route table contains an entry for that name.

2. The names appearing in APPLN records are not entered into the selection screen presented to users accessing PVMG through VTAM.

AUTHORIZ



Purpose

Use the AUTHORIZ record to identify users who are authorized to issue the restricted subset of PVMG commands and the nodes from which these commands may be issued.

Operands

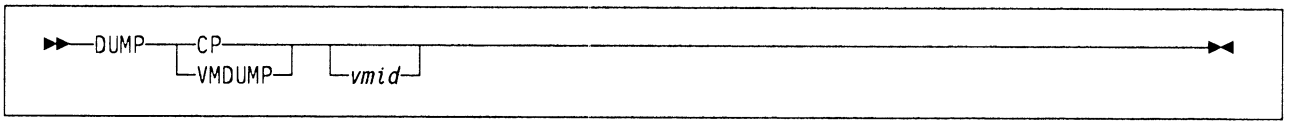
vmid
is the identification of a VM user (or virtual machine) authorized to issue restricted PVMG commands. Without specifying * or *nodeid*, this authorization is for the local node only.

SYSTEM
authorizes the PVMG operator to issue commands from nodes (* or *nodeid*) other than the local node.

* authorizes the user to issue restricted PVMG commands *from any node.*

nodeid
is a specific node where *vmid* is authorized to issue restricted PVMG commands.

DUMP



Purpose

Use the DUMP record to indicate whether the dump taken during abnormal termination of PVMG is a CP dump or a VM IPCS dump.

Operands

CP
indicates that on abnormal termination of PVMG the dump is CP DUMP.

VMDUMP
indicates that on abnormal termination of PVMG the dump is VMDUMP.

vmid
indicates where the dump is to be spooled. The default is written to the printer for a CP dump and spooled to the *vmid* of IPCS for a VMDUMP.

Usage Notes

1. The PVM/IPCS Extension and the VM/ESA and VM/XA Dump Viewing Facility require a VMDUMP type dump. Therefore, VMDUMP *must* be specified if you use either PVM/IPCS or the VM/ESA and VM/XA Dump Viewing Facility.
2. If the DUMP record is omitted, the default setting is DUMP CP. If more than one DUMP record is entered in the PVMG configuration file, only the last DUMP record in the list is accepted. PVMG will ignore all other DUMP records.

LANG

Purpose

Use the LANG record to identify a particular language as the default national language for a node.

During PVMG initialization, the backup national language is considered the national language for a node before the LANG record is processed. It continues as the default national language for the node if the LANG configuration file record is not specified.

Operands

langid
is the identification of the selected language. It is a 1-to-5-character language identifier.

The IBM-provided languages are:

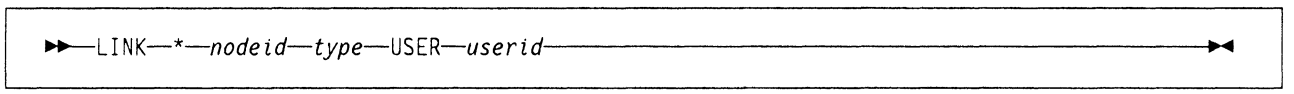
AMENG American English

KANJI Japanese

UCENG Uppercase English

GER German

LINK



Purpose

Use the LINK record to define the link for connecting with a PVM virtual machine only (PVMG cannot link to another PVMG virtual machine through an IUCV connection). Use separate LINK records for each PVM virtual machine to which you wish to connect.

The same node identifier cannot appear in more than one LINK record.

Note: It is very important that the *nodeid* specified in the LINK record be a name that is unique throughout the entire PVM network.

Operands

* indicates that a line address is not provided.

nodeid
is the node name of the PVM virtual machine to which an IUCV connection is to be provided.

type
specifies the type of line driver used to drive the line. The only valid type for PVMG is:

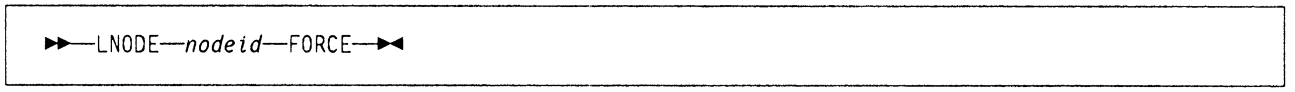
IUCV uses the IUCV protocol to communicate with the PVM virtual machine.

USER *userid*
indicates that a userid for the PVM virtual machine follows. The *userid* is a 1-to-8-character string identifying the PVM virtual machine to which the IUCV connection is to be made.

Usage Notes

1. If the link on the PVM side is not active, PVMG will attempt the IUCV connect at 60 second intervals until either communications are established, or a PVMG DROP node command is issued by an operator.

LNODE



Purpose

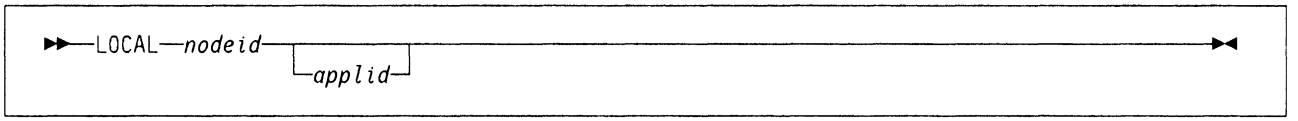
Use the LNODE record to define the node name that is automatically selected when a user accesses PVMG from an SNA network, bypassing the PVM Selection screen. If you want to go to a different system than the one specified, you must pass a node name using NODE=*nodeid* as part of the DATA option on the VTAM LOGON command. If any other options are passed using the DATA option on the VTAM LOGON command, they are acted upon by PVMG, but no messages will be issued for errors that are enountered.

Operands

nodeid
is the name of the default selection node.

FORCE
forces a session with *nodeid*.

LOCAL



Purpose

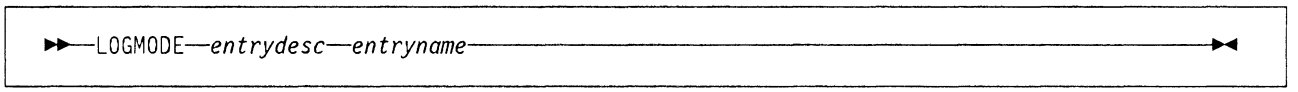
Use the LOCAL record to define the node by which the PVMG server is known to other nodes in the network, and the application name by which VTAM recognizes PVMG.

Operands

nodeid
is the name of the local node. The *nodeid* can be 1 to 8 characters. Default is a reserved word and should not be used as the *nodeid*.

applid
is the name by which PVMG will be known to VTAM. The *applid* can be 1 to 8 characters. If not specified, the *nodeid* is used.

LOGMODE



Purpose

Use the LOGMODE record to specify VTAM logon mode table entries that are used by PVMG when establishing interactive sessions from PVM to the SNA network.

Operands

entrydesc
describes the VTAM Logon Mode Table entry. The *entrydesc* must be in the VTAM logmode table. The following operand descriptions are supported:

entrydesc	**Operand Description**
3277	indicates that the Logon Mode Table Entry Name is for a 3277 terminal
3278-2B	indicates that the Logon Mode Table Entry Name is for a 3278 model 2 terminal
3278-2E	indicates that the Logon Mode Table Entry Name is for a 3278 model 2 terminal with extended features
3278-3B	indicates that the Logon Mode Table Entry Name is for a 3278 model 3 terminal
3278-3E	indicates that the Logon Mode Table Entry Name is for a 3278 model 3 terminal with extended features
3278-4B	indicates that the Logon Mode Table Entry Name is for a 3278 model 4 terminal

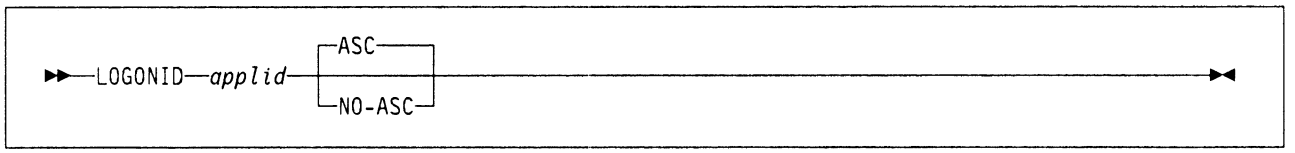
entrydesc	**Operand Description**
3278-4E	indicates that the Logon Mode Table Entry Name is for a 3278 model 4 terminal with extended features
3278-5B	indicates that the Logon Mode Table Entry Name is for a 3278 model 5 terminal
3278-5E	indicates that the Logon Mode Table Entry Name is for a 3278 model 5 terminal with extended features
3279-2B	indicates that the Logon Mode Table Entry Name is for a 3279 model 2 terminal
3279-2E	indicates that the Logon Mode Table Entry Name is for a 3279 model 2 terminal with extended features
3279-3B	indicates that the Logon Mode Table Entry Name is for a 3279 model 3 terminal
3279-3E	indicates that the Logon Mode Table Entry Name is for a 3279 model 3 terminal with extended features
3290	indicates that the Logon Mode Table Entry Name is for a 3290 terminal. The terminal is treated as a 3278 Model 2.

entryname
specifies the VTAM Logon Mode Table entry name that applies to the *entrydesc* parameter.

Usage Notes

1. PVMG looks for each of the *entrydesc*s listed in the preceding table during configuration file processing. If one is missing, a message is issued, and PVMG terminates after configuration file processing is complete.

LOGONID



Purpose

Use the LOGONID record to define the VTAM APPLID to which the PVMG node is to log the PVM terminals onto when the PVMG node is selected.

Operands

applid
is the name of the application, SAMON, for example, on to which PVMG is to log PVM terminals.

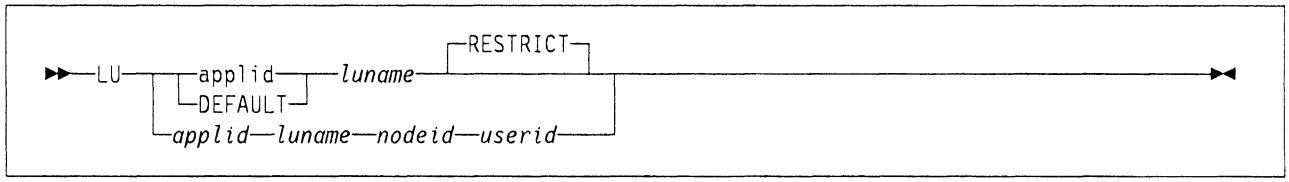
ASC
specifies that the application is an Application Selection Concentrator. This is the default if a second operand is not provided.

NO-ASC
specifies that the application is not an Application Selection Concentrator.

Usage Notes

1. When SNA users issue the VTAM LOGON command to access a PVM network, the *applid* supplied is the one specified on the PVMG LOCAL configuration file record, not the *applid* on this record.

LU



Purpose

Use the LU record to specify LU names that are to be used by PVMG when obtaining interactive VTAM sessions with restricted and secure VTAM APPLIDs. When a PVM user selects an application for a session, PVMG assigns an LU name from the list of LU names in the LU record. If the application has the RESTRICT option, then an available LU is assigned. If the application requires specific users, then the user making the session request must be assigned an LU name in the LU record.

Operands

applid
is the VTAM APPLID. The *applid* can be 1 to 8 characters.

DEFAULT
the *DEFAULT* operand defines *lunames* to be used for applications that do not have specific *lunames* defined for them. All *LU DEFAULT* records must precede any other LU records for restricted VTAM applications.

luname
is the name of the SNA logical unit associated with the *applid*. The *luname* may be a single name, or it may be a range of SNA logical unit names. The range represents a pool of logical unit names that may be used to access the specified *applid*. If the optional *nodeid* and *userid* are encoded in this record, *luname* must be a single LU. If the third parameter is RESTRICT or omitted, then the second operand can be a range.

The format of a range is given by:

- A base name that also is the lower boundary of the range
- An upper boundary of the range
- A hyphen (-) separates the lower from the upper boundary of the range.

For example,

```
SAM05000-99
```

will result in the generation of 100 logical units with addresses SAM05000, SAM05001,...SAM05099.

```
SAM05050-99
```

will result in the generation of 50 logical units with addresses SAM05050, SAM05051,...SAM05099.

The following rules apply to ranges:

- The upper boundary must be all digits, and must not exceed a length of three.
- The number of trailing numeric digits in the base must be greater than or equal to the number of numeric digits in the range. For example, the range specification

  ```
  SAM050A0-99
  ```

  is not acceptable since the trailing two characters in the base, A0, are not all numeric. The specification

  ```
  SAM05A20-100
  ```

  is unacceptable since the trailing three characters in the base, A20, are not all numeric.
- The range of logical unit names that can be generated from a single statement cannot exceed 500.

RESTRICT
indicates that only one session may be assigned to one LU. This is the default if no option is specified.

nodeid
is the nodeid to which this LU is secured.

userid
is the userid to which this LU is secured.

Usage Notes

1. Unless an application requires specific LU names, or must be secure, one or more LU DEFAULT configuration file records should suffice for your installation needs.
2. The same LU name, or range of LU names, can be specified on multiple restricted LU records for different APPLIDs. However, each LU name can be used by only one user to a single application at a time.
3. An application cannot be both secure and restricted, but an LU name can be used for secure, as well as restricted, applications.
4. If an APPLID must be secure for multiple users, each secure LU record must have a unique LU name associated with it.
5. If you specify a restricted LU, only one user can use an LU at a time. If you specify an range of restricted LUs in the form:

   ```
   LU XXX01 XXX0001-50 RESTRICT
   ```

   50 LUs will be set up to handle 50 users at once, each user with their own LU. This saves entering 50 individual LU records.
6. If

   ```
   LU XXX01 XXX0001 RMTPVM1 JONES
   ```

   is entered, only user JONES at RMTPVM1 may use the specified LU.

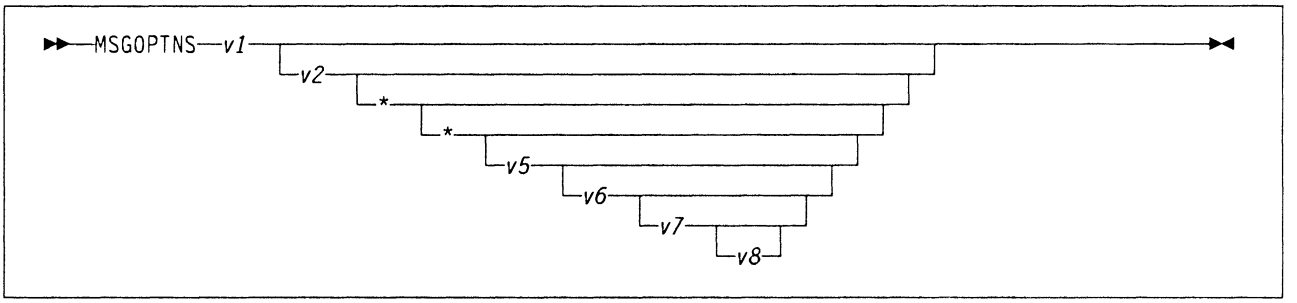
MSGNOH

Purpose

Use the MSGNOH record to indicate that responses to all commands entered by the CP SMSG command must be returned using the CP MSGNOH command rather than the CP MSG command, unless prohibited by the MSGOPTNS on that record when sending messages to the CP OPERATOR. If you enter more than one MSGNOH record in the configuration file, only the last MSGNOH record in the list is accepted. PVMG ignores the other MSGNOH records.

PVMG must have Class B privilege to issue the CP MSGNOH command.

MSGOPTNS



Purpose

Use the MSGOPTNS record to indicate what options PVMG will use regarding time stamps and message numbers for messages directed to the PVMG user ID, to users, and to the operator. The installation can choose for each destination whether time stamps and message numbers should be appended to the messages. All parameters (except 3 and 4) must be either a Y for yes or an N for no.

The MSGOPTNS record does not support an audit file, but is compatible with the PVM component's MSGOPTNS record.

Operands

v1 specifies the time stamps on the messages to the PVMG user ID. The default is N.

v2 specifies the message numbers on the messages to the PVMG user ID. The default is N.

* reserved (use * as a placeholder). PVMG will ignore any non-blank character(s) used as a placeholder. A placeholder is used to ensure consistency with the PVM MSGOPTNS configuration file record.

* reserved (use * as a placeholder). PVMG will ignore any non-blank character(s) used as a placeholder. A placeholder is used to ensure consistency with the PVM MSGOPTNS configuration file record.

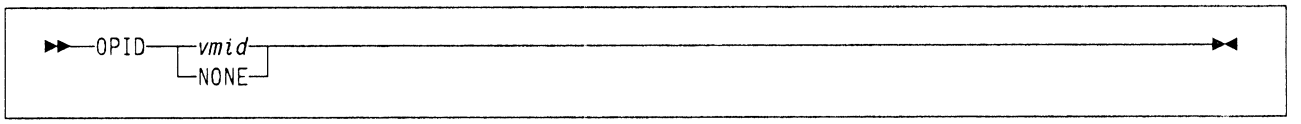
v5 specifies the time stamps on the messages to users. The default is N.

v6 specifies the message numbers on the messages to users. The default is N.

v7 specifies all messages written to the operator. These messages are written by a CP MSG command, overriding the MSGNOH record. The default is N, meaning that the MSGNOH option will take effect.

v8 specifies the message numbers on the messages to the operator. The default is N.

OPID



Purpose

Use the OPID record to specify the virtual machine ID that receives messages that normally go to the system operator. You can also use this record to stop messages from being sent to the system operator.

Operands

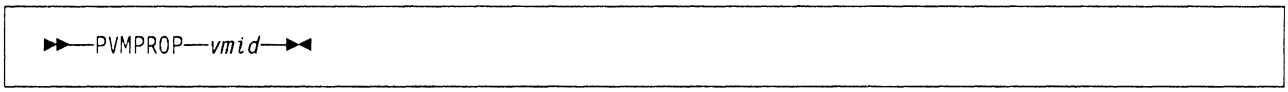
vmid
is the identification of the virtual machine to receive operator messages.

NONE
suppresses the messages sent to the operator.

Usage Notes

1. If you omit this record, operator messages are sent to the OPERATOR user ID.
2. The format of the messages sent to the specified operator depends on the options specified on the MSGOPTNS record and on whether PVMG is authorized to use the CP MSGNOH command.
3. Only one *vmid* may be specified on this record.

PVMPROP



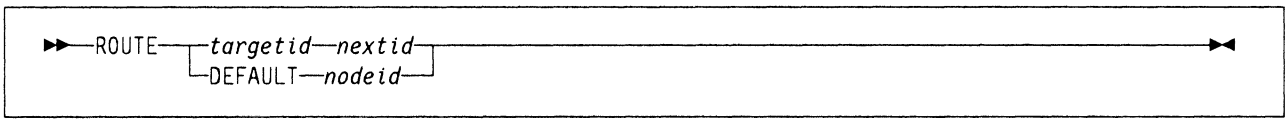
Purpose

Use the PVMPROP record to identify a virtual machine running a programmed operator that receives a copy of PVMG error messages. All messages are transmitted to the virtual machine by the CP Special Message (CP SMSG) Facility. The messages are sent to the PVMPROP virtual machine in the same encoded format as sent to the Command Response Interface. See Chapter 15, "Facilities for Automating PVM and PVMG Operations" on page 251 for more information about PVMPROP.

Operands

vmid
is the virtual machine to receive error messages. All PVMG messages are sent to the *vmid* in encoded form using the CP SMSG command.

ROUTE



Purpose

Use the ROUTE record to define the next nodal path to a specified target node from the local PVMG node. For PVMG to PVM communications, routing must be consistently defined on both ends of network paths.

Use the second form of the ROUTE record to define default routing for nodes not defined by LINK or ROUTE records. You can enter only one default route record.

Operands

targetid
is the identification of the target system node.

nextid
is the identification of the next PVM node defined in a configuration file LINK record, in the routing pattern from the local node to the target system node.

DEFAULT *nodeid*
indicates that the *nodeid* is the default node identifier. The *nodeid* is the PVM node through which all unknown nodes are to be routed. This node must be defined in a LINK configuration file record. To avoid difficulties, you should not define any PVM or PVMG node identifiers with the name *DEFAULT*.

SELECT

Purpose

Use the SELECT record to assign a frequently used node name to a PF key. You can specify up to 11 SELECT records to assign 11 PF keys The first SELECT record *nodeid* is assigned to PF1, the second SELECT record *nodeid* is assigned to PF2, and so forth to PF7. The eighth SELECT record is assigned to PF9 (PF8 is reserved for the SCROLL FORWARD function). The ninth SELECT record is assigned to PF10, and so forth to PF12.

If you use a SELECT record without specifying the node ID, the PF key associated with this SELECT record is omitted. This allows a PF key to be skipped.

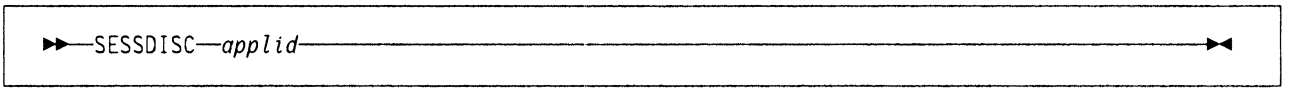
Operands

nodeid
is the identification of the node assigned to the PF key represented by this SELECT record. The *nodeid* may be 1 to 8 characters long.

Usage Notes

1. Two SELECT records in the configuration file cannot contain the same node name.

SESSDISC



Purpose

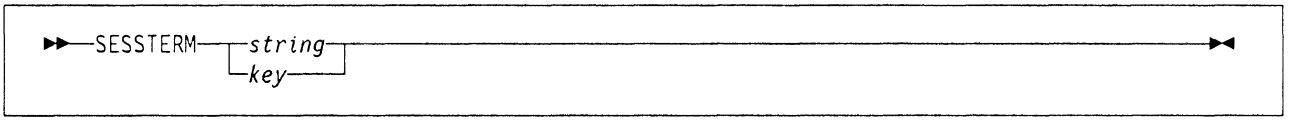
Use the SESSDISC record to indicate the VTAM APPLID to which PVMG should issue the VTAM TERMSESS macro with the optional parameter OPTCD=COND, when unexpected PVM session failures occur. If an unexpected PVM session failure occurs for an APPLID not specified on the SESSDISC record, PVMG issues the VTAM TERMSESS macro with OPTCD=UNCOND, to terminate the session.

The specific action can occur only when PVMG executes as a secondary logical unit. You should determine what session termination actions are provided by an application as the primary logical unit before deciding to specify its APPLID on this record.

Operands

applid
is the name of the VTAM application. The *applid* can be 1 to 8 characters.

SESSTERM



Purpose

Use the SESSTERM record to define the default session termination string, *PFn* or *PA* key used at the local PVMG node for VTAM originating users.

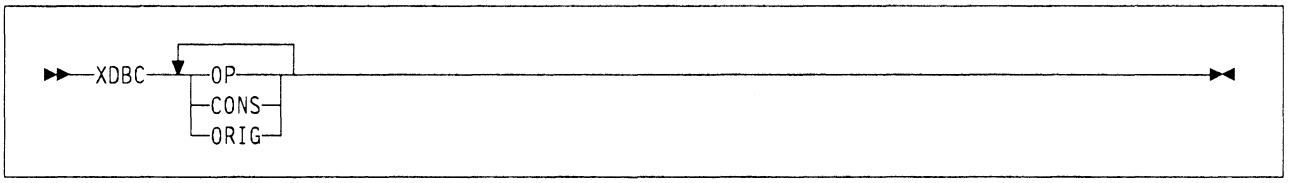
If this record is not entered in the PVMG configuration file, PVMG uses the default string ####.

Operands

string
is a 1-to-8-character, session termination string. Mixed case strings are allowed.

key
is the key you wish to assign for a session termination key. Use the form *PFn* or *PAn* to specify the key. For PF keys, *nn* can be a number from 01 (or 1) to 24. For PA keys, *n* can be 1 or 2.

XDBC



Purpose

Use the XDBC record to exclude certain users on the local system from receiving, in the national language, any PVMG messages that, in the national language, contain double-byte characters. The identified users receive those messages in the backup language instead. The record may appear anywhere in the file.

The principal reason for the XDBC record is that PVMG cannot tell whether the virtual machine consoles of other virtual machines on a VM system are capable of handling double-byte characters.

Operands

OP
identifies the CP operator.

CONS
identifies the PVMG user ID console (display station).

ORIG
identifies all virtual machine users on the local VM system who are not PVMG users, other than the CP operator and the PVMG operator. That is, the originators of actions or commands to which PVMG messages are sent in response.

If the LANG record identifies a DBCS language and the PVMG user ID or the CP operator should not receive DBCS messages, the XDBC record should be coded immediately before or after the LANG record.

If PVMG finds an error in the XDBC record, PVMG immediately excludes OP, CONS, and ORIG from receiving DBCS messages, and it is necessary to use the CDBC command to enable them again.

Appendix G. Setting Up an APPC Link

This appendix describes how to set up an APPC link. The following are sample CP directories, CMS communication directories, configuration files, and PVM profile execs for two PVM nodes, for example, RMTPVM1 and RMTPVM5.

You can use these partial samples as models when you set up an APPC link.

SYSTEM -- RMTPVM1

SMITH is a userid on RMTPVM1.

Sample CP Directory for the PVM user ID

```
USER PVM XXXXXXX 5M 8M BG 64
OPTION   BMX MAXCONN 4096 ECMODE
IPL CMS
IUCV ALLOW
IUCV *IDENT RESANY GLOBAL
```

Sample CMS Communications Directory entry for RMTPVM1 (UCOMDIR NAMES)

```
:nick.EAGLE    :luname.PVMGATE1 PVMGATE5
               :tpn.PVM5
               :modename.AVSMODE
               :security.PGM
               :userid.JONES
               :password.IANIAN
```

Partial Configuration File for RMTPVM1 (PVM CONFIG)

```
⋮
LOCAL RMTPVM1
RESID PVM1
LINK EAGLE RMTPVM5 APPC USER SMITH
⋮
```

Partial PVM Profile Exec for RMTPVM1 (PROFILE PVM)

```
⋮
START NODE RMTPVM5
⋮
```

SYSTEM -- RMTPVM5

JONES is a userid on RMTPVM5.

Sample CP Directory for the PVM user ID

```
USER PVM XXXXXXX 5M 8M BG 64
OPTION   BMX MAXCONN 4096 ECMODE
IPL CMS
IUCV ALLOW
IUCV *IDENT RESANY GLOBAL
```

Sample CMS Communications Directory entry for RMTPVM5 (UCOMDIR NAMES)

```
:nick.HAWK     :luname.PVMGATE5 PVMGATE1
               :tpn.PVM1
               :modename.AVSMODE
               :security.PGM
               :userid.SMITH
               :password.WALTER
```

Partial Configuration File for RMTPVM5 (PVM CONFIG)

```
⋮
LOCAL RMTPVM5
RESID PVM5
LINK HAWK RMTPVM1 APPC USER JONES
⋮
```

Partial PVM Profile Exec for RMTPVM5 (PROFILE PVM)

```
⋮
START NODE RMTPVM1
⋮
```

The following are explanations of the information used in the samples above:

- USER PVM, in the CP directory for PVM on RMTPVM1, identifies PVM as the user ID for that PVM virtual machine.

  USER PVM, in the CP directory for PVM on RMTPVM5, identifies PVM as the user ID for that PVM virtual machine.

- EAGLE, the :nick. entry in UCOMDIR NAMES for RMTPVM1, is a symbolic destination name of the node you are connecting to, RMTPVM5. EAGLE corresponds to the parameter on the LINK record in the configuration file for RMTPVM1.

© Copyright IBM Corp. 1991,1993

HAWK, the :nick. entry in UCOMDIR NAMES for RMTPVM5, is a symbolic destination name of the node you are connecting to, RMTPVM1. HAWK corresponds to the parameter on the LINK record in the configuration file for RMTPVM5.

- PVM1GATE and PVM5GATE are the gateway names of the local system and the system to which you are connecting. PVM1GATE is the gateway name for RMTPVM1, and PVM5GATE is the gateway name for RMTPVM5. These names are the names known to the SNA network.
- PVM5 is the resource name of PVM on system RMTPVM5, which is identified by the RESID record in the configuration file for RMTPVM5. PVM5 is the resource name RMTPVM1 wants to connect to, which is identified by the :tpn. entry in UCOMDIR NAMES for RMTPVM1. If the RESID record is omitted, the resource name would have to match the name on the LOCAL record.

  Likewise, PVM1 is the resource name of PVM on system RMTPVM1, which is identified by the RESID record in the configuration file for RMTPVM1. PVM1 is the resource name RMTPVM5 wants to connect to, which is identified by the :tpn. entry in UCOMDIR NAMES for RMTPVM5. If the RESID record is omitted, the resource name would have to match the name on the LOCAL record.

  Each resource name must be unique within a PVM network.
- AVSMODE is the mode name. This name is found in the VTAM logon mode table. See your VTAM administrator for the mode name.
- PGM is a keyword identifying the security level. PGM or SAME must be specified.
- JONES, the :userid. entry in UCOMDIR NAMES for RMTPVM1, is a valid user ID on RMTPVM5. This user ID matches the user ID on the LINK record in the configuration file for RMTPVM5.

  SMITH, the :userid. entry in UCOMDIR NAMES for RMTPVM5, is a valid user ID on RMTPVM1. This user ID matches the user ID on the LINK record in the configuration file for RMTPVM1.
- IANIAN is the password of JONES at RMTPVM5.

  WALTER is the password of SMITH at RMTPVM1.

Appendix H. PVM Control Block Formats

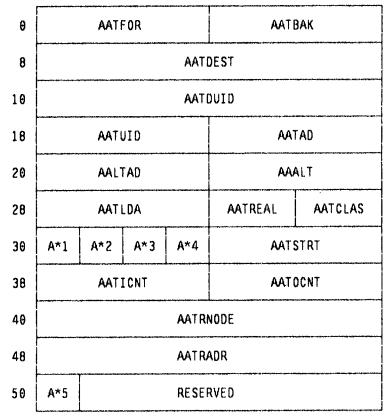
The information in this appendix is included to provide further information to help you take advantage for the sample access security exits, and for diagnostic purposes.

This appendix contains Product Sensitive Programming Interface information.

PSPI

DVMAAT (Active Application Task Descriptor) Format

Each DVMAAT control block describes an active logical device support task in the PVM (for sessions in which the local PVM is the target) virtual machine.

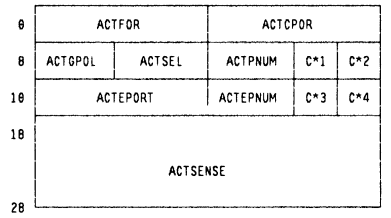


Field	Length	Use
AATFOR	4	FORWARD POINTER CHAIN
AATBAK	4	BACKWARD POINTER CHAIN
AATDEST	8	DESTINATION NODE ID
AATDUID	8	DESTINATION USER ID
AATUID	4	TASK USERID (PRINTABLE LOGICAL DEVICE ADDRESS)
AATAD	4	ADDRESS OF OWNING TASK (DIAGNOSE DRIVER) TCB
AALTAD	4	ADDRESS OF TCB FOR TASK WHICH IS TO GET DATA BUFFERS
AAALT	4	ADDRESS OF ALT ENTRY
AATLDA	4	LOGICAL DEVICE ADDRESS
AATREAL	2	REAL DEVICE ADDRESS
AATCLAS	2	DEVICE CLASS AND TYPE
A*1 (AATMDL)	1	DEVICE MODEL NUMBER
A*2 (AATTYPE)	1	TYPE FLAGS
	Values defined in AATTYPE:	
AATEXTN	X'80'	REMOTE END HAS EXTENDED ATTRIBUTES
AATIOV	X'40'	USER WANTS I/O VERIFICATION
A*3 (AATRFEA)	1	EXTENDED FEATURES
A*4 (AATSTYPE)	1	SESSION TYPE
AATSTRT	4	SESSION START TIME
AATICNT	4	INPUT TRANSACTIONS
AATOCNT	4	OUTPUT TRANSACTIONS
AATRNODE	8	ORIGIN NODE IF R3270
AATRADR	8	EBCDIC REAL TERMINAL ADDRESS
A*5 (AATMARG)	1	MESSAGE ARGUMENT BYTE
	7	RESERVED

© Copyright IBM Corp. 1991,1993

DVMACT (Cluster Table Entry) Format

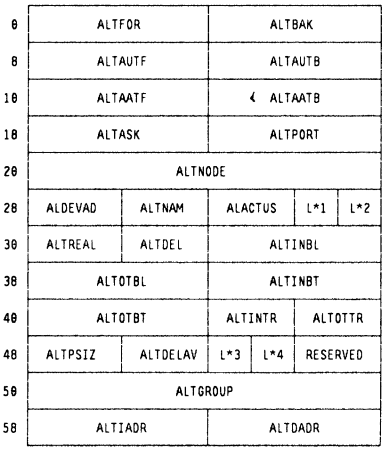
Each DVMACT control block describes an individual cluster on an R3270 control unit link as defined by a CLUSTER configuration file record.



Field	Length	Use
ACTFOR	4	FORWARD POINTER CHAIN
ACTCPOR	4	ADDRESS OF PORT LIST
ACTGPOL	2	GENERAL POLLING ADDRESS
ACTSEL	2	SELECTION CONTROL UNIT ADDRESS
ACTPNUM	2	NUMBER OF PORTS SUPPORTED
C*1 (ACTCNUM)	1	CONTROL UNIT NUMBER
C*2 (ACTSTAT)	1	CONTROL UNIT STATUS FLAGS
	Values defined in ACTSTAT:	
	ACTSXTN	X'80' CONTROL UNIT IS 3274-1C WITH EXTENDED FEATURES
	ACTSON	X'40' CONTROL UNIT IS ONLINE
	ACTSOFF	X'20' CONTROL UNIT VARIED OFFLINE
	ACTASGN	X'10' RE-CHECK THE CLUSTER
	ACTRSEL	X'08' CLUSTER IN RE-SELECT
	ACTERR1	X'04' CLUSTER IN ERROR RECOVERY
ACTEPORT	4	ADDRESS OF PORT IN ERROR
ACTEPNUM	2	NUMBER OF PORT IN ERROR
C*3 (ACTECNT)	1	ERROR RETRY COUNT
C*4 (ACTRSV1)	1	RESERVED
ACTSENSE	16	SENSE BIT COUNTERS

DVMALT (Active Link Table Entry) Format

Each DVMALT control block defines a link (line driver) to another node.

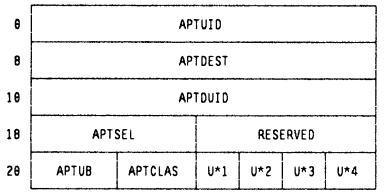


Field	Length	Use
ALTFOR	4	FORWARD CHAIN POINTER
ALTBAK	4	BACKWARD CHAIN POINTER
ALTAUTF	4	CHAIN OF USER TASKS
ALTAUTB	4	CHAIN OF USER TASKS
ALTAATF	4	CHAIN OF APPL TASKS
ALTAATB	4	CHAIN OF APPL TASKS

Field			Length	Use
ALTASK			4	ADDRESS OF OWNING TASK TCB
ALTPORT			4	ADDRESS OF PORT LIST
ALTNODE			8	NODE NAME
ALDEVAD			2	TP LINE ADDRESS
ALTNAM			2	LINK TYPE
ALACTUS			2	NUMBER OF ACTIVE SESSIONS
L*1 (ALTFLG)			1	FLAGS FOR LINK
	Values defined in ALTFLG:			
	ALTCON	X'80'		LINK WAS OR IS CONNECTED - DETERMINES WHETHER TO RESTART IF LINE FAILS
	ALTACTV	X'40'		LINEDRIVER TASK IS STARTED
	ALTDOWN	X'20'		INDICATES A DROP HAS BEEN ISSUED
	ALTQUIES	X'10'		THE LINE IS QUIESCING
	ALTRSTRT	X'08'		INDICATES NETWORK LINE RESTARTING
L*2 (ALTTPRMS)			1	TRACING FLAGS
	Values defined in ALTTPRMS:			
	ALTTRIO	X'80'		I/O TRACING IS ACTIVE
	ALTTRDAT	X'40'		DATA TRACING IS ACTIVE
	ALTTRWRP	X'20'		USE IN-CORE WRAP TABLE
ALTREAL			2	REAL LINE ADDRESS
ALTDEL			2	LINE DELAY IN MILLISECONDS
ALTINBL			4	TRANSMISSION BLOCKS RECEIVED
ALTOTBL			4	TRANSMISSION BLOCKS SENT
ALTINBT			4	BYTES RECEIVED
ALTOTBT			4	BYTES TRANSMITTED
ALTINTR			2	EXTRA RECORDS PER BLOCK SENT
ALTOTTR			2	EXTRA RECORDS PER BLOCK RECEIVED
ALTPSIZ			2	BLOCKSIZE THRESHOLD
ALTDELAV			2	AVERAGED LINK DELAY TIME
L*3 (ALTOPN1)			1	LINK CLASS OPTIONS
	Values defined in ALTOPN1:			
	ALTOTBND	X'80'		OUT-BOUND SESSIONS ALLOWED
	ALTINBND	X'40'		IN-BOUND SESSIONS ALLOWED
	ALTNTIME	X'20'		LINK SUPPORTS LINK TIMING
	ALTOGRP	X'10'		LINK SUPPORTS LINE GROUPING
L*4 (ALTOPTN2)			1	LINK FEATURE OPTIONS
	Values defined in ALTOPTN2:			
	ALTDIA98	X'80'		DIAG 98 WANTED FOR LINK
	ALTPACKF	X'40'		PACKING IS REQUESTED
	ALTNADDR	X'20'		LINK ADDRESS NOT TO BE CHECKED
	ALTNDEF	X'10'		LINK NOT VALID FOR DEFINE COMMAND
	ALTNMOD	X'08'		LINK TYPE CANNOT BE ALTERED
			2	RESERVED
ALTGROUP			8	GROUP NAME IF PART OF LINE GROUP
ALTIADR			4	ADDRESS OF I/O TRACE BUFFER
ALTDADR			4	ADDRESS OF DATA TRACE BUFFER

DVMAPT (Active Remote Port Descriptor) Format

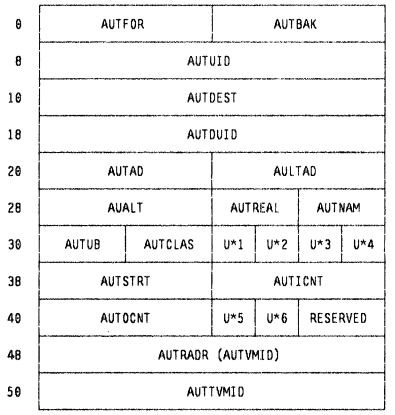
Each DVMAPT control block contains session information for a cluster port on an R3270 remote control unit link that currently is engaged with an active session.



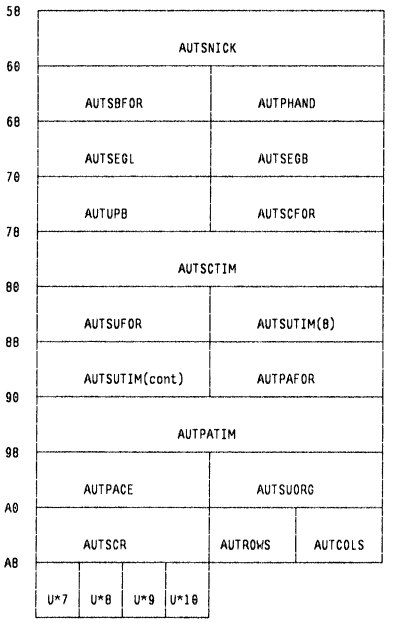
Field	Length	Use
APTUID	8	USER ID
APTDEST	8	DESTINATION NODE ID
APTDUID	8	DESTINATION USER ID
APTSEL	4	SELECT SCREEN BUFFER
RESERVED	4	RESERVED
APTUB	2	TUBE DEVICE ADDRESS
APTCLAS	2	REAL DEVICE CLASS AND TYPE
U*1 (APTMDL)	1	REAL DEVICE MODEL NUMBER
U*2 (APTTYPE)	1	TYPE OF INTERFACE TASK
	Values defined in APTTYPE:	
	APTEXTN X'80'	TERMINAL SUPPORTS EXTENSION FEATURE
U*3 (APTSTAT)	1	STATUS OF INTERFACE TASK
	Values defined in APTSTAT:	
	APTDSEL X'80'	SELECTION SCREEN WAS USED
	OCBUFSW X'40'	ORDERS CHECK PANEL DISPLAYED
U*4 (APTUFEA)	1	EXTENDED FEATURES BEING USED
	Values defined in APTUFEA:	
	VIRTSHI X'10'	HIGHLIGHT NOW INFORCE FOR GUEST
	VIRTSCOL X'20'	COLOR NOW INFORCE FOR GUEST
	VIRTSPSS X'40'	PSS NOW INFORCE FOR GUEST

DVMAUT (Active User Interface Link Descriptor) Format

Each DVMAUT control block describes an active user support task in this PVM.



Each DVMAUT extension area for the MPVM user support task only applies to an individual MPVM session.



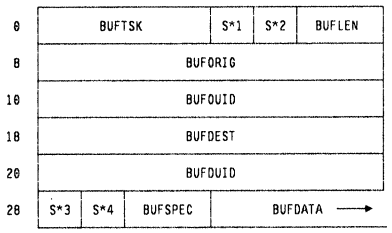
Field	Length	Use
AUTFOR	4	FORWARD CHAIN POINTER
AUTBAK	4	BACKWARD CHAIN POINTER
AUTUID	8	USER ID
AUTDEST	8	DESTINATION NODE ID
AUTDUID	8	DESTINATION USER ID
AUTAD	4	ADDRESS OF OWNING TASK (INTERFACE DRIVER) TCB
AULTAD	4	ADDRESS OF TCB FOR TASK WHICH IS TO GET DATA BUFFERS
AUALT	4	ADDRESS OF ALT ENTRY
AUTREAL	2	REAL DEVICE ADDRESS
AUTNAM	2	MNEMONIC LINK TYPE
AUTUB	2	TUBE DEVICE ADDRESS
AUTCLAS	2	REAL DEVICE CLASS AND TYPE
U*1 (AUTMDL)	1	REAL DEVICE MODEL NUMBER

Field		Length	Use
U^2 (AUTTYPE)		1	TYPE OF INTERFACE TASK
	Values defined in AUTTYPE:		
	AUTGRAF	X'80'	DIRECT GRAPHIC INTERFACE
		X'40'	RESERVED
	AUTAAT	X'20'	REMOTE END IS A LOGICAL DEVICE
	AUTIUCV	X'10'	IUCV SESSION
	AUTEXTN	X'08'	DISPLAY HAS EXTENDED FEATURES
		X'04'	RESERVED
	AUTPRT	X'02'	THIS IS A PRINTER TASK
		X'01'	RESERVED
U^3 (AUTSTAT)		1	STATUS OF INTERFACE TASK
	Values defined in AUTSTAT:		
	AUTWAIT	X'80'	AWAITING SESSION INITIATION
	AUTSEL	X'40'	DISPLAYING SELECTION
	AUTCONN	X'20'	SESSION IS ACTIVE
	AUTNRDY	X'10'	PRINTER IN NOT READY STATE
	AUTSGNON	X'04'	IN PROCESS OF REMOTE SIGNON
	AUTIOV	X'02'	USER WANTS I/O VERIFICATION
	AUTOFLG	X'01'	THIS IS ORIGIN OF SESSION
U^4 (AUTRFEA)		1	REAL EXTENDED FEATURES
	Values defined in AUTRFEA:		
	AUTRECOL	X'80'	DEVICE HAS EXTENDED COLOR
	AUTREHLT	X'40'	DEVICE HAS EXTENDED HIGHLIGHTING
	AUTRPSS	X'20'	DEVICE HAS PROGRAMMED SYMBOL SETS
	AUTDBCS	X'10'	DEVICE SUPPORTS DOUBLE BYTE CHARACTERS
AUTSTRT		4	SESSION START TIME (TOP TOD)
AUTICNT		4	INPUT TRANSACTION COUNT
AUTOCNT		4	OUTPUT TRANSACTION COUNT
U^5 (AUTSTYPE)		1	SESSION TYPE
U^6 (AUTMARG)		1	MESSAGE ARGUMENT BYTE
		2	RESERVED
AUTRADR (AUTVMID)		8	EBCDIC REAL TERMINAL ADDRESS OR IUCV ORIGIN VMID
AUTTVMID		8	IUCV DESTINATION VMID OR MPVM USER'S VMID
AUTSNICK		8	THE MPVM USER'S SESSION NICKNAME
AUTSBFOR		4	NEXT MPVM USER SESSION BLOCK
AUTPHAND		4	MPVM'S SESSION HANDLE
AUTSEGL		4	SEGMENTED DATA LENGTH
AUTSEGB		4	SEGMENTED DATA BUFFERS QUEUE
AUTUPB		4	SCREEN UPDATE BUFFERS QUEUE
AUTSCFOR		4	NEXT TIMER IN CHAIN AFTER SCREEN UPDATE
AUTSCTIM		8	SCREEN UPDATE TIMER VALUE AND CODE
AUTSUFOR		4	NEXT TIMER IN CHAIN AFTER SUSPEND TIMER
AUTSUTIM		8	SUSPEND TIMER VALUE AND CODE
AUTPAFOR		4	NEXT TIMER IN CHAIN AFTER PACING TIMER
AUTPATIM		8	PACING TIMER VALUE AND CODE
AUTPACE		4	PACING OVERFLOW QUEUE OF BUFFERS
AUTSUORG		4	ORIGINAL SUSPEND TIME VALUE WHEN FIRST SET
AUTSCR		4	ADDRESS OF COMPRESSED BUFFER OR SCREEN IMAGE DSECT
AUTROWS		2	NUMBER OF ROWS ON USER'S SCREEN
AUTCOLS		2	NUMBER OF COLUMNS ON USER'S SCREEN
U^7 (AUTPCNT)		1	BUFFERS SINCE LAST PACING
U^8 (AUTPNUM)		1	UNANSWERED PACING COUNT
U^9 (AUTSES1)		1	MPVM USER FLAGS
	Values defined in AUTSES1:		
	AUTPAC	X'80'	PACING ACTIVE
	AUTWFSC	X'40'	WAITING FOR SESSION TO COMPLETE OR FOR A TERMINATE RECORD
	AUTSUS	X'20'	SESSION IS SUSPENDED
	AUTST	X'10'	SCREEN FOR SESSION BEING TRANSFERRED
	AUTTERM	X'08'	SESSION BEING TERMINATED
	AUTSS1	X'04'	FIRST SUSPEND WARNING SET
	AUTSS2	X'02'	SECOND SUSPEND WARNING SENT

Field		Length	Use
	AUTPCR	X'01'	POSSIBLE CHAINED READ
U^10 (AUTSES2)		1	MPVM USER FLAGS
	Values defined in AUTSES2:		
	AUTTDISC	X'80'	USER AND SESSION ARE TEMPORARILY DISCONNECTED
	AUTXPAND	X'40'	SCREEN FOR THIS SESSION IS EXPANDED
	AUTNONOT	X'20'	NO NOTIFICATION TO MPVM USER FOR SESSION TERMINATE
	AUTREPLY	X'10'	NEED TO DO AN IUCV REPLY TO MPVM
	AUTNET	X'08'	NET WILL HANDLE SESSION NOW
	AUTPRLD	X'04'	PREVIOUS LOAD/COMPRESS PAIR DONE ON CURRENT SCREEN IMAGE

DVMBUF (Internal Buffer Header)

This data area contains a description of the header portion of the buffered data that is transferred between tasks within the PVM network.

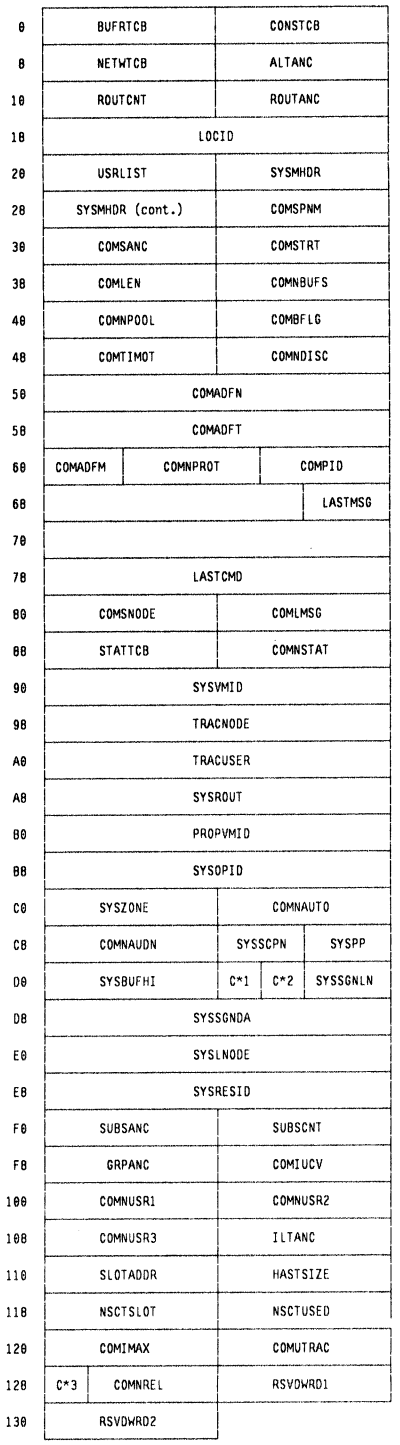


Field		Length	Use
BUFTSK		4	TCB ADDRESS OF OWNING TASK
S^1 (BUFTYPE)		1	RECORD TYPE
	Values defined in BUFTYPE:		
	BUFTCTL	X'80'	NETWORK CONTROL RECORD
	BUFTIMER	X'40'	LINK TIMING RECORD
	BUFTGDAT	X'20'	GRAPHIC DATA RECORD
	BUFTDATA	X'10'	NON-GRAPHIC DATA RECORD
	BUFTCMD	X'04'	LOCAL COMMAND BUFFER
S^2 (BUFSTYP)		1	RECORD SUBTYPE
	Values defined in BUFSTYP when BUFTYPE= X'80':		
	BUFSNET	X'20'	NETWORK PATH CONTROL
	BUFSIGN	X'10'	SESSION INITIATE REQUEST
	BUFSGNR	X'08'	SESSION COMPLETE
	BUFSOFF	X'04'	SESSION TERMINATION
	BUFCMSG	X'02'	'CMD', 'MSG', AND REPLIES
	Values defined in BUFSTYP when BUFTYPE= X'40':		
	BUFRETRN	X'40'	THIS IS THE RETURN BUFFER
	Values defined in BUFSTYP when BUFTYPE= X'20':		
	GRAFRDBF	X'20'	DATA WAS RETURNED FROM A READ-BUFFER COMMAND
	GRAFMSG	X'10'	MESSAGE TEXT TO BE DISPLAYED ON THE SCREEN
	GPRTSTAT	X'08'	STATUS RETURNED FROM I/O
	GSRDBUF	X'04'	SHORT READ-BUFFER REQUEST
	GSTATRQ	X'02'	STATUS MUST BE RETURNED
	GCHNCCW	X'01'	PART OF COMMAND CHAIN SEQUENCE
	Values defined in BUFSTYP when BUFTYPE= X'10' and bit 0 of BUFFLG1= 0:		
	BUFRECV	X'20'	IUCV REPLY
	Values defined in BUFSTYP when BUFTYPE= X'10': and bit 0 of BUFFLG1= 1:		
	BUFRJCT	X'40'	IUCV REJECT OF TWO-WAY SEND
	BUFRECV	X'20'	IUCV REPLY
	BUFPRTY	X'10'	DATA IS PRIORITY DATA
	BUFPMSG	X'08'	DATA IS VIA PRMDATA
	BUF0QUIS	X'04'	THIS IS AN IUCV QUIESCE
	BUF0RESU	X'04'	THIS IS AN IUCV RESUME
BUFLEN		2	LENGTH OF DATA IN BUFFER
BUFORIG		8	ORIGIN NODE ID

Field		Length	Use
BUFOUID		8	ORIGIN USER ID
BUFDEST		8	DESTINATION NODE ID
BUFDUID		8	DESTINATION USER ID
S^3 (BUFFLG1)		1	FLAG BYTE
	Values defined in BUFFLG1:		
	BUFSFORM	X'80'	NEW FORMAT SESSION PACKET
	BUFDIRF	X'40'	PACKET IS FROM SESSION ORIGIN
	BUFPACE	X'20'	PACING FLAG
	BUFAPAC	X'10'	RESPONSE TO PACING
	BUFSEGF	X'0F'	NUMBER OF 256 BYTE SEGMENTS IN BUFFER
	Values for commands and messages:		
	BUFNLSR4	X'80'	THE ORIGIN IS RELEASE 4
	BUFENCOD	X'40'	BUFFER CONTAINS AN ENCODED MESSAGE
S^4 (BUFFLG2)		1	SPANNED RECORD FLAGS
	Values defined in BUFFLG2:		
	BUFSFLG	X'80'	THIS IS A SPANNED RECORD
	BUFSPFE	X'40'	THIS IS THE LAST RECORD SEGMENT
	BUFSCNT	X'3F'	SPANNED RECORD NUMBER 0-63
BUFSPEC		2	SPECIAL DEFINITION
	Values defined in BUFSPEC for COMMAND packets (BUFTYPE= X'80' and BUFSTYP= X'02'):		
	SYSCMD	X'40'	COMMAND FROM CONSOLE
	CMDRPLY	X'20'	REPLY TO COMMAND
	CPCMD	X'10'	CP COMMAND
	SYSUSER	X'08'	SYSTEM USER
	CMDSMSG	X'04'	REPLY BY SMSG
	BUFSIGNA	X'02'	SIGNATURE SET IN BUFTAG
	BUFSUBST	X'01'	SPECIAL SUBSET COMMAND
BUFDATA		—	START OF DATA AREA

DVMCOMN (PVM Multitasker Common Area) Format

The DVMCOMN control block contains information commonly required by PVM tasks.



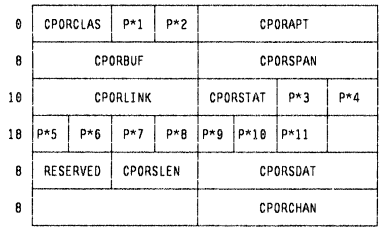
Field	Length	Use
BUFRTCB	4	BUFFER TCB ADDRESS
CONSTCB	4	CONSOLE INPUT TCB ADDRESS
NETWTCB	4	NETWORK ROUTER TCB ADDRESS
ALTANC	4	ALT ANCHOR
ROUTCNT	4	NUMBER OF ENTRIES IN ROUT TABLE
ROUTANC	4	ADDRESS OF ROUTE TABLE

Field	Length	Use
LOCID	8	LOCAL NODE ID
USRLIST	4	ADDRESS OF USER CONNECT LIST
SYSMHDR	8	CP MESSAGE COMMAND OP CODE
COMSPNM	4	NUMBER OF ENTRIES IN SYSTEM PROGRAMMER LIST
COMSANC	4	ADDRESS OF SYSTEM PROGRAMMER LIST
COMSTRT	4	ADDRESS OF START OF TABLE AREA
COMLEN	4	LENGTH OF TABLE AREA
COMNBUFS	4	NUMBER OF BUFFER PAGES
COMNPOOL	4	ADDRESS OF BUFFER POOL START
COMBFLG	4	ADDRESS OF BUFFER FLAG TABLE
COMTIMOT	4	TP LINE TIMEOUT CHECK TIME
COMNDISC	4	USER TEMP DISCONNECT TIME
COMADFN	8	FILENAME OF AUDIT FILE
COMADFT	8	FILETYPE OF AUDIT FILE
COMADFM	2	FILEMODE OF AUDIT FILE
COMNPROT	3	LINE DRIVER PROTOCOL LEVEL
COMPID	9	COMPONENT ID
LASTMSG	10	LAST LOCAL PVM MESSAGE ISSUED
LASTCMD	8	LAST PVM COMMAND ISSUED
COMSNODE	4	POINTER TO LOCAL NODES
COMLMSG	4	POINTER TO LOGMSG
STATTCB	4	TCB OF STATISTICS TASK (DVMSTA)
COMNSTAT	4	STATISTICS TIME INTERVAL
SYSVMID	8	SYSTEM VMID
TRACNODE	8	TRACE NODENAME
TRACUSER	8	TRACE USERID
SYSROUT	8	DEFAULT LINK ROUTING
PROPVMID	8	DEFAULT VMID FOR PVMPROP
SYSOPID	8	DEFAULT VMID FOR OP MESSAGES
SYSZONE	4	TIME ZONE OFFSET IN MINUTES
COMNAUTO	4	AUTO EXEC TIME INTERVAL
COMNAUDN	4	MAXIMUM RECORDS FOR AUDIT FILE
SYSSCPN	2	SYSTEM NAME FROM DIAG 0
SYSPP	2	FIRST 16 BITS OF PP FLAGS
SYSBUFHI	4	HIGH BUFFER PAGE ALLOCATED
C*1 (COMNFLG1)	1	MESSAGE OPTIONS FLAGS
	Values defined in COMNFLG1:	
	CFLG1CT	X'80' CONSOLE - TIMESTAMP
	CFLG1CM	X'40' CONSOLE - MESSAGE NUMBER
	CFLG1AT	X'20' AUDIT - TIMESTAMP
	CFLG1AM	X'10' AUDIT - MESSAGE NUMBER
	CFLG1UT	X'08' USER - TIMESTAMP
	CFLG1UM	X'04' USER - MESSAGE NUMBER
	CFLG1ON	X'02' OPID - MSG, NOT MSGNOH
	CFLG1OM	X'01' OPID - MESSAGE NUMBERS
C*2 (SYSCOMPT)	1	COMPATIBILITY LEVEL
SYSSGNLN	2	SESSION TERMINATE LENGTH
SYSSGNDA	8	SESSION TERM STRING
SYSLNODE	8	LOCAL NODE SELECT
SYSRESID	8	LOCAL RESOURCE ID
SUBSANC	4	ANCHOR FOR SUBSET COMMANDS
SUBSCNT	4	NUMBER OF SUBSET ENTRIES
GRPANC	4	ANCHOR FOR LINE GROUPING
COMIUCV	4	ADDRESS OF IUCV SLOT TABLE
COMNUSR1	4	USER AREA
COMNUSR2	4	USER AREA
COMNUSR3	4	USER AREA
ILTANC	4	ANCHOR FOR PCCF TASKS
SLOTADDR	4	ADDRESS OF HASH SLOT TABLE
HASTSIZE	4	SIZE IN BYTES OF SLOT TABLE
NSCTSLOT	4	NUMBER OF 6-BYTE SLOTS

Field	Length	Use
NSCTUSED	4	NUMBER OF USED SLOTS
COMIMAX	4	MAXIMUM NUMBER OF IUCV/APPC PATHS
COMUTRAC	4	ADDRESS OF USER TRACING BUFFER
C^3 (COMNVER)	1	PVM VERSION NUMBER
COMNREL	3	PVM RELEASE, SERVICE LEVEL
RSVDWRD1	4	RESERVED FULL WORD
RSVDWRD2	4	RESERVED FULL WORD

DVMCPOR (3271-4 Cluster Port Entry) Format

Each DVMCPOR control block describes a physical cluster port on an R3270 remote control unit link as defined by a CLPORT configuration file record.

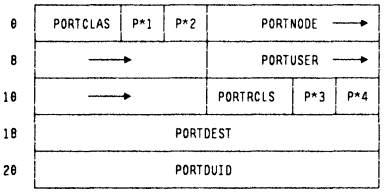


Field		Length	Use
CPORCLAS		2	DEVICE CLASS AND TYPE
P^1 (CPORMDL)		1	DEVICE MODEL
P^2 (CPORFLG1)		1	FLAG FIELD
	Values defined in CPORFLG1:		
	CPORVALD	X'80'	PORT IS DEFINED
	CPORACTV	X'40'	PORT IS ACTIVE
	CPORSEL	X'20'	SELECTION SCREEN DISPLAYED
	CPORSESS	X'10'	PORT IN SESSION
	CPORLCL	X'08'	LOCAL SESSION WANTED
	CPORSGN	X'04'	WAITING FOR SESSION
	CPORASGN	X'02'	RE-INIT NEEDED
	CPOROFF	X'01'	VARIED OFFLINE
CPORAPT		4	ADDRESS OF DVMAPT
CPORBUF		4	ADDRESS OF OUTPUT BUFFER
CPORSPAN		4	ADDRESS OF SPANNED RECORD CHAIN
CPORLINK		4	ADDRESS OF OPTIONAL PARMS
CPORSTAT		2	3270 REMOTE STATUS
P^3 (CPORECNT)		1	ERROR RETRY COUNT
P^4 (CPORFLG2)		1	FLAG FIELD
	Values defined in CPORFLG2:		
	CPORPACE	X'80'	PACING IS ACTIVE
	CPORAAT	X'40'	SESSION WITH LOGICAL DEVICE
	CPORMSG	X'20'	PVM MESSAGE PROCESSING
	CPORSXTN	X'10'	3270 WITH EXTENDED FEATURES
	CPORWSF	X'08'	PROCESSING A WSF GENERAL QUERY
	CPORSPSW	X'04'	PROCESSING A SPANNED RECORD
	CPORREL1	X'02'	CONNECTED TO RELEASE 1 NODE
	CPORBUSY	X'01'	DEVICE IS BUSY
P^5 (CPORCPAC)		1	BUFFERS BETWEEN PACING
P^6 (CPORNPAC)		1	UNANSWERED PACING
P^7 (CPORLEW)		1	LAST ERASE WRITE COMMAND
P^8 (CPORRFEA)		1	REAL DEVICE EXTENDED FEATURES
	Values defined in CPORRFEA:		
	REALECOL	X'80'	DEVICE HAS EXTENDED COLOR
	REALEHLT	X'40'	DEVICE HAS EXTENDED HIGHLIGHTING
	REALPSS	X'20'	DEVICE HAS PROGRAMMING SYMBOL SETS
	REALDBCS	X'10'	DEVICE SUPPORTS DBCS SHIFT CHARACTERS
P^9 (CPORRSRM)		1	REPLY MODE SUPPORTED

Field		Length	Use
P*10 (CPORFLG3)		1	FLAG FIELD
	Values defined in CPORFLG3:		
	CPORIOV	X'80'	I/O VERIFICATION WANTED
	CPORSTRQ	X'40'	I/O STATUS TO BE RETURNED
	CPOR7GA	X'20'	3277 GRAPHICS ATTAHCED
	CPORDBSY	X'10'	DEVICE BUSY STATUS RETURNED
P*11 (CPORMARG)		1	LANGUAGE ARGUMENT BYTE
CPORHOT		1	HOT I/O COUNTER
		2	RESERVED
CPORSLEN		4	LENGTH OF SESSION TERMINATE STRING
CPORSDAT		8	SESSION TERMINATE STRING
CPORCHAN		4	QUEUE OF COMMAND CHAINED BUFFERS

DVMPORT (Simulator Link Port Table Entry)

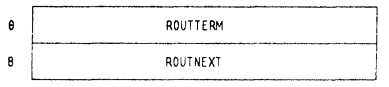
Each DVMPORT control block describes a particular port on an emulator link as defined by a PORT configuration file record.



Field		Length	Use
PORTCLAS		2	DEVICE CLASS AND TYPE
P*1 (PORTMDL)		1	DEVICE MODEL
P*2 (PORTFLG1)		1	FLAG FIELD
	Values defined in PORTFLG1:		
	PORTVALD	X'80'	THIS IS A VALID PORT
	PORTINUS	X'40'	PORT IS ALLOCATED
	PORTRSVD	X'20'	PORT IS RESERVED
	PORTSDE	X'10'	LOGOFF DEVICE END TO BE SENT
	PORTUSR	X'08'	PORT RESERVED FOR USER
	PORTWACK	X'04'	SEND WACK ON SELECT, PTR ONLY
PORTNODE		8	AUTHORIZED NODE
PORTUSER		8	AUTHORIZED USER
PORTRCLS		2	REAL DEVICE CLASS AND TYPE
P*3 (PORTRMDL)		1	REAL DEVICE MODEL NUMBER
P*4 (PORTTYPE)		1	TYPE FIELD FROM DVMAUT
PORTDEST		8	DESTINATION NODE
PORTDUID		8	DESTINATION USER ID

DVMROUT (Entry in Route List) Format

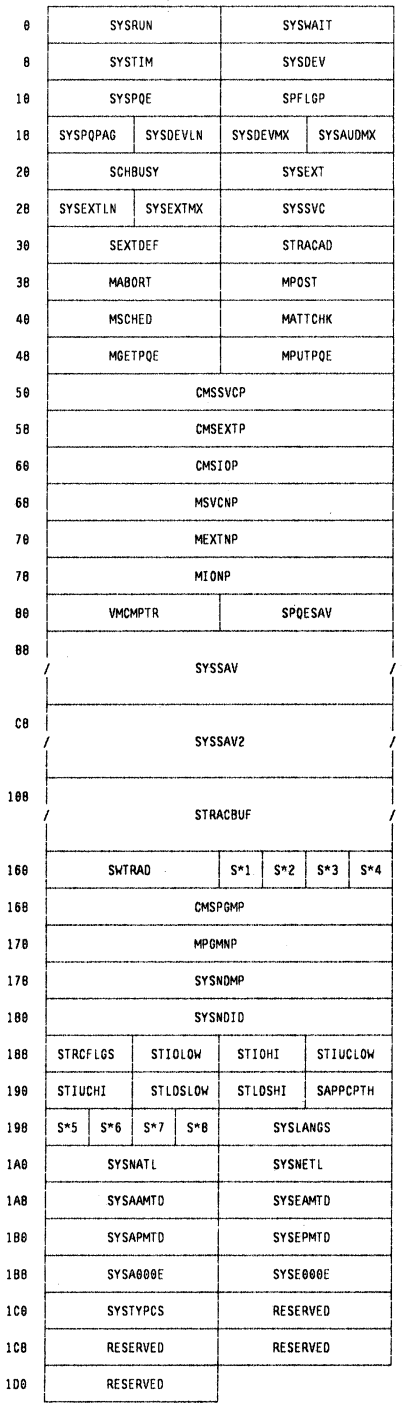
Each DVMROUT control block identifies the target node, followed by the node through which the routing occurs.



Field	Length	Use
ROUTTERM	8	TARGET NODE NAME
ROUTNEXT	8	NEXT NODE IN LINE

DVMSYSB (PVM Multitasker System Control Block) Format

DVMSYSB is the main communication control block for the multitasking supervisor.



Field	Length	Use
SYSRUN	4	RUN Q ANCHOR
SYSWAIT	4	WAIT Q ANCHOR
SYSTIM	4	TIMER Q ANCHOR
SYSDEV	4	POINTER TO SYSTEM DEVICE TABLE
SYSPQE	4	POINTER TO PQE POOL
SPFLGP	4	POINTER TO PQE POOL FLAGS
SYSPQPAG	2	NUMBER OF PQE PAGES
SYSDEVLN	2	DEVICE TABLE CURRENT ENTRY COUNT
SYSDEVMX	2	MAXIMUM DEVICE TABLE ENTRY COUNT
SYSAUDMX	2	MAX AUDIT FILE RECORD LENGTH
SCHBUSY	4	CHANNEL BUSY BITS FOR IOS
SYSEXT	4	POINTER TO EXT HANDLE TABLE
SYSEXTLN	2	CURRENT EXTTAB ENTRY COUNT
SYSEXTMX	2	MAXIMUM EXTTAB ENTRY COUNT
SYSSVC	4	POINTER TO SVC TABLE
SEXTDEF	4	TCB ADDRESS FOR EXT DEFAULTS
STRACAD	4	ADDRESS OF TRACE ROUTINE
MABORT	4	ADDRESS OF ABORT ROUTINE
MPOST	4	ADDRESS OF POST ROUTINE
MSCHED	4	ADDRESS OF SCHEDULER
MATTCHK	4	ADDRESS OF CHECK IF ATTACHED ROUTINE
MGETPQE	4	ADDRESS OF PQE GET ROUTINE
MPUTPQE	4	ADDRESS OF PQE PUT ROUTINE
CMSSVCP	8	CMS SVC NEW PSW
CMSEXTP	8	CMS EXTERNAL NEW PSW
CMSIOP	8	CMS I/O NEW PSW
MSVCNP	8	MULTITASKING SUPERVISOR SVC NEW PSW
MEXTNP	8	MULTITASKER EXTERNAL NEW PSW
MIONP	8	MULTITASKER I/O NEW PSW
VMCMPTR	4	POINTER TO VMCF INTERRUPT ELEMENT
SPQESAV	4	POINTER TO SAVED PQE
SYSSAV	64	REGISTER SAVE AREA FOR SUPERVISOR ROUTINES
SYSSAV2	64	ANOTHER REGISTER SAVE AREA FOR SUPERVISOR ROUTINES
STRACBUF	88	TRACE BUFFER
SWTRAD	4	ADDRESS OF OUTPUT WRITER TASK
S^1 (SYSFLAG)	1	SYSTEM FLAGS
	Values defined in SYSFLAG:		
	SYSETRAC	X'80'	TRACE USING ETRACE
	SYSCTRAC	X'40'	TRACE TO VIRTUAL CONSOLE
	SYSPTRAC	X'20'	TRACE TO PRINTER (00F)
	SYSTRACE	X'E0'	ANY TRACING IS ACTIVE
	SMSSSW	X'10'	USE ^MSS FACILITY
	SYSTRAP	X'08'	ENABLE TRAP CODE
	SYSSHUT	X'04'	SYSTEM IS SHUTTING DOWN
	SYSQUIS	X'02'	SYSTEM IS IN A QUIESCE STATE
	SYSAUDIT	X'01'	LOG THE CONSOLE
S^2 (SYSFLAG2)	1	SYSTEM FLAGS	
	Values defined in SYSFLAG2:		
	SAUTONOD	X'80'	FORCE AUTO-NODE SELECT
	SRESUM	X'40'	TASK-MUST-COMPLETE FLAG
	SYSDISC	X'20'	CONSOLE IS DISCONNECTED
	SRECMS	X'10'	TASK RESUME FROM CMS
	SYSXA	X'04'	VM/XA RUNNING
	SYSXASP	X'02'	VM/XA SP RUNNING

Field		Length	Use
S*3 (SYSFLAG3)		1	SYSTEM FLAGS
	Values define in SYSFLAG3:		
	SIUCVSW	X'80'	IUCVIS ALLOWED
	SDIAG98	X'40'	DIAG 98 IS ALLOWED
	SLDSCSW	X'20'	CP INDICATE READ CHAINED TO WRITE
	SIUCLIST	X'10'	LIST FORM ALLOWED WITH IUCV
	S327EDS	X'08'	CP HAS EXTENDED LSDF
	SRESMGR	X'04'	PVM is *IDENT RESOURCE MANAGER
	SRESCON	X'02'	SUCCESSFUL IUCV CONNECT *IDENT
	SNAPPC	X'01'	SYSTEM CAN'T HANDLE APPC
S*4 (SYSXDBC)		1	DBCS EXCLUSIONS FLAGS FOR PSWS
CMSPGMP		8	CMS PGM NEW PSW
MPGMNP		8	MULTITASKER PROGRAM NEW PSW
SYSNDMP		8	TYPE OF DUMP
SYSNDID		8	VMID FOR DUMP
STRCFLGS		2	TRACING FLAGS
	Values defined in STRCFLGS:		
	SDISPTR	X'80'	TRACE TASK DISPATCHING
	SPOSTTR	X'40'	TRACE POSTING OF TASKS
	SIOTR	X'20'	TRACE START I/O's AND I/O INTERRUPTS
	SVMCFTR	X'10'	TRACE VMCF DIAGNOSE AND INTERRUPTS
	SLDSFTR	X'08'	TRACE LSDF DIAGNOSE AND INTERRUPTS
	SIUCVTR	X'04'	TRACE IUCV INSTRUCTIONS AND INTERRUPTS
	STIMRTR	X'02'	TRACE TIMER REQUESTS AND INTERRUPTS
	SSTORTR	X'01'	TRACE STORAGE ALLOCATIONS
	Values defined in STRCFLGS+1:		
	SATTCTR	X'80'	TRACE ATTACH AND DETACH
	SSVCTR	X'40'	TRACE SVC INSTRUCTIONS
	SDSTOTR	X'20'	TRACE DATA BUFFER STORAGE ALLOCATION
STIOLOW		2	LOW LIMIT FOR I/O TRACING
STIOHI		2	HIGH LIMIT FOR I/O TRACING
STIUCLOW		2	LOW LIMIT FOR IUCV TRACING
STIUCHI		2	HIGH LIMIT FOR IUCV TRACING
STLDSLOW		2	LOW LIMIT FOR LDSF TRACING
STLDSHI		2	HIGH LIMIT FOR LDSF TRACING
SAPPCPTH		2	IUCV PATHID TO LOCAL *IDENT RESIDE
S*5 (SYSFLAG5)		1	SYSTEM FLAGS
	Values defined in SYSFLAG5:		
	SDIAGEC	X'80'	SYSTEM SUPPORTS DIAG X'EC'
	SISFSW	X'40'	SYSTEM SUPPORTS ISF
S*6 (SYSFLAG6)		1	RESERVED
S*7 (SYSFLAG7)		1	RESERVED
S*8 (SYSFLAG8)		1	RESERVED
SYSLANGS		4	BASE OF ENABLED LANGUAGE LIST
SYSNATL		4	NATIONAL LANGUAGE BASE ADDRESS
SYSNETL		4	BACKUP LANGUAGE BASE ADDRESS
SYSAAMTD		4	NATIONAL LANGUAGE A.M. TIMESTAMP
SYSEAMTD		4	BACKUP LANGUAGE A.M. TIMESTAMP
SYSAPMTD		4	NATIONAL LANGUAGE P.M. TIMESTAMP
SYSEPMTD		4	BACKUP LANGUAGE P.M. TIMESTAMP
SYSA000E		4	ABR's 000E NATIONAL MESSAGE ADDRESS
SYSE000E		4	ABR's 000E NETWORK MESSAGE ADDRESS
SYSTYPCS		4	10, 50, 68, & 127 TYPE CHARACTERS
RESERV1		4	RESERVED
RESERV2		4	RESERVED
RESERV3		4	RESERVED
RESERV4		4	RESERVED

DVMTCB

Each task in PVM has a task control block (DVMTCB) with which it is associated. The DVMTCB has two parts:

1. A common area common to each task, and
2. An extension area that is defined by, and used exclusively by, each task.

The format of the common area is:

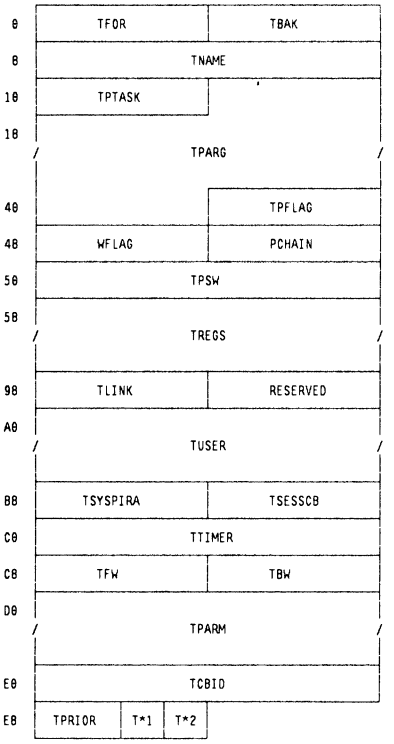


Figure 41. DVMTCB Control Block Format

Field	Length	Use
TFOR	4	FORWARD CHAIN POINTER
TBAK	4	BACKWARD CHAIN POINTER
TNAME	8	LINK NAME
TPTASK	4	TCB ADDRESS OF POSTING TASK
TPARG	48	POSTED ARGUMENT
TPFLAG	4	FLAG FOR CURRENT POST
WFLAG	4	WAIT ENABLED FLAGS
PCHAIN	4	POSTED ELEMENT CHAIN (contents may change whenever system is enabled)
TPSW	8	OLD PSW FOR THIS TASK
TREGS	64	REGISTER SAVE AREA FOR THIS TASK
TLINK	4	ADDRESS OF THE LINK CONTROL BLOCK FOR THIS TASK
RESERVED	4	RESERVED
TUSER	24	USER AREA
TSYSPIRA	4	ADDRESS OF PROGRAM CHECK INTERRUPT ROUTINE ADDRESS
TSESSCB	4	ADDRESS OF SESSION CONTROL BLOCK
TTIMER	8	CLOCK VALUE FOR TIMER INTERRUPT
TFW	4	TIMER FORWARD CHAIN
TBW	4	TIMER BACKWARD CHAIN
TPARM	16	ARGUMENTS TO MULTITASKING SUPERVISOR SERVICE ROUTINES
TCBID	8	TASK ID
TPRIOR	2	TASK SCHEDULING PRIORITY

Field		Length	Use
T^1 (TATTR)		1	TASK ATTRIBUTES
	Values defined in TATTR:		
	TDMSFR	X'80'	MUST DMSFRE TCB WHEN DONE
	TWAIT	X'40'	TASK NOW WAITING
	TPEND	X'20'	TASK EXECUTION PENDING
	TIOW	X'10'	TASK WAITING - SIO BUSY
T^2		1	RESERVED

The format of the TCB extension for the network session manager (DVMNET) is:

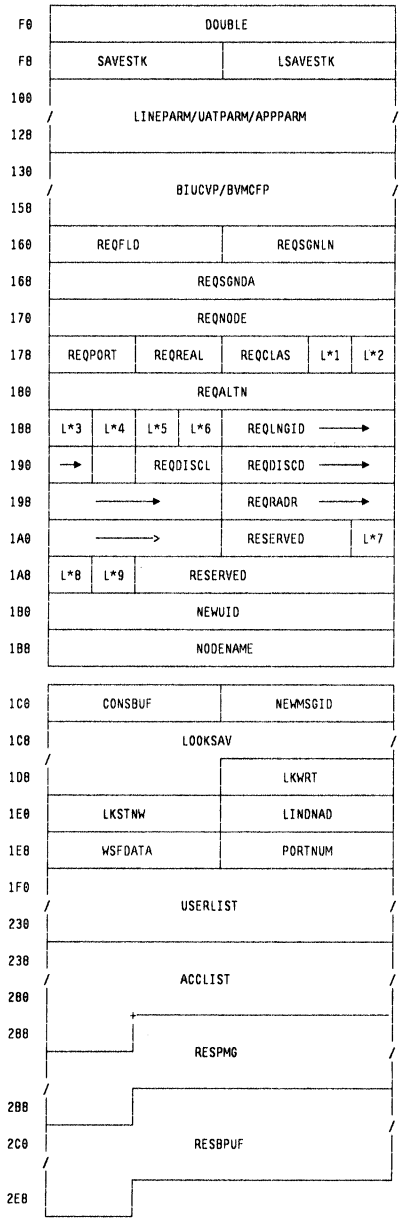


Figure 42. DVMNTCB Control Block Format

Field	Length	Use
DOUBLE	8	WORK AREA
SAVESTK	4	POINTER TO STACK AREA
LSAVESTK	4	LENGTH OF STACK IN DWs
LINEPARM/ UATPARM/ APPPARM	48	TASK ATTACH PARAMETER AREA

Field	Length	Use
BIUCVP/BVMCFD	48	IUCV OR VMCF PARAMETER AREA.
REQFLD	4	CONTAINS WORD 'NETW'
REQSGNLN	4	SESSION DISC LENGTH
REQSGNDA	8	SESSION DISC DATA
REQNODE	8	ANY SPECIFIC NODE
REQPORT	2	PORT NUMBER OR X'FF'
REQREAL	2	TERMINAL ADDRESS
REQCLAS	2	TERMINAL CLASS
L^1 (REQMDL)	1	TERMINAL MODEL NUMBER
L^2 (REQTYPE)	1	ATTRIBUTE FLAGS
Values defined in REQTYPE:		
REQEXTN	X'80'	TERMINAL HAS EXTENDED FEATURES
REQDBCS	X'01'	TERMINAL SUPPORTS DBCS
REQALTN	8	ALTERNATE ROUTING
L^3 (REQFLGS)	1	SPECIALTY FLAGS
Values defined in REQFLGS:		
REQIOV	X'80'	I/O VERIFICATION REQUESTED
L^4 (REQMARG)	1	NLS COMPATIBILITY LEVEL
L^5 (REQLNGNM)	1	NLS LANGUAGE NUMBER
Values defined in REQLNGNM:		
REQLNGEX	X'01'	USER CHOSE AN EXPLICIT LANGUAGE
L^6 (REQNPKEY)	1	NOTEPAD KEY
REQLNGID	5	NLS LANGUAGE NAME
	1	RESERVED
REQDISCL	2	TEMPORARY DISCONNECT LENGTH
REQDISCD	8	TEMPORARY DISCONNECT STRING/KEY
REQRADR	8	EBCDIC TERMINAL ADDRESS
RESERVED	3	RESERVED
L^7 (SCAN)	1	SCAN NUMBER
L^8 (UMDL)	1	DISPLAY MODEL
L^9 (PMDL)	1	PORT MODEL
RESERVED	6	RESERVED
NEWUID	8	NEW USERID
NODENAME	8	SAVE AREA FOR NODE ID
CONSBUF	4	ADDRESS OF CONSOLE COMMAND BUFFER
NEWMSGID	4	MESSAGE I.D. FROM NEW REQUEST
LOOKSAVE	20	SAVE AREA FOR LOOKAUT ROUTINE
LKWRT	4	LINK SAVE AREA FOR WRTCON ROUTINE
LKSTNW	4	LINK SAVE AREA FOR START NEW ROUTINE
LINDNAD	4(	SAVE FOR TCB TO GO TO FREE STORE ... WHEN RECYCLING LINE DRIVER
WSFDATA	4	ADDRESS OF WRITE STRUCTURED FIELD DATA
PORTNUM	4	SAVE AREA FOR PORT NUMBER
USERLIST	64	USER EXIT PARAMETER LIST
ACCLIST	82	STATISTICS EXIT PARAMETER LIST
RESPMSG	48	RESPONSE MESSAGE AREA
RESPBUF	48	RESPONSE MESSAGE AREA

PSPI end

Appendix I. Manually Dialing a Remote 4300 Processor

Note: This procedure requires a telephone dial up to do this manual dial. If you are not an operator with access to the dial equipment, then contact the operator who can make the call.

1. Turn on the display station and wait for the VM logo.
2. When the VM logo appears, enter `dial pvm` on the command line:

```
USERID   ===>
PASSWORD ===>

COMMAND  ===> dial pvm
```

   where `pvm` is the name of the PVM user ID.

3. When the Pass-Through Facility Selection Menu appears, select the appropriate ROCF node. This message:

```
DIAL
```

   indicates that the Remote Operations Communication Facility (ROCF) is ready and can accept a manual dial.

4. Perform the manual dialing of the remote ROCF system. You must initiate a telephone dial connection to the remote node.
5. A message is sent to the display:

```
CONNECT
```

   This message is immediately replaced by the next panel transmitted from the ROCF node.

6. The display of a 4300 ROCF Logon/Password panel signifies that the connection to the remote 4300 processor is complete.
7. Enter the password.
8. The 4300 General Selection panel is displayed. This panel offers a menu of available options. Choose the Program Load (L) option. Enter:

   `L`

   following the SELECTION label. The Program Load panel appears. This panel controls the processor IML and system IPL.

9. If the Program Load panel indicates that the IML is complete, then you may invoke an IPL by:

1. Typing the required IPL UNIT ADDR parameters on the panel
2. Typing `P` following the SELECTION label
3. Pressing ENTER.

The Status area on the Program Load panel displays:

```
IPL COMPLETE
```

© Copyright IBM Corp. 1991,1993

indicating the IPL is complete.

1. End the PVM session by entering:

   ####

   PVM terminates the dialed connection to the ROCF node. Prior to entering the Session Terminate string, you had the option to return to the programming system to perform VM system functions.

2. The Pass-Through Facility Selection Menu appears with a message in the message line:

   ```
   SESSION ENDED BY USER
   ```

   indicating the PVM session with the ROCF system has ended. After receiving this message, press the PA1 key to indicate that no further PVM work is desired.

Appendix J. PVM Access Security Exits

DMTI

This chapter describes the security exits sample code provided for the PVM component. These security exit routines will enable an installation to control sessions originating at a local node, outside the local node, and commands which are destined for or originated at another node. This will be accomplished by using a table called REMOTE AUTH which must be located on a minidisk that is accessed by PVM's user ID. These security routines do not restrict Cross-Systems Extension (CSE) applications.

Notes:

1. To determine if a session initiation request record is from a CSE application, the new format session packet bit in BUFFLG1 and the format for the CSE request bit (FORM3) in SCTLFORM must both be ON. See "DVMBUF (Internal Buffer Header)" on page 425 for information the DVMBUF format.
2. The REMOTE AUTH file can also have a file name of the PVM local node id. The access security exits looks for file *nodeid* AUTH first and if not found, then REMOTE AUTH. This file will be referred to as REMOTE AUTH throughout this appendix.
3. These exits are included in the PVM 2.1.1 module supplied by IBM on the product tape and built by VMSES/E. If an installation does not require the use of these exits, then place a dummy REMOTE AUTH file on PVM's 191 minidisk or a disk the PVM user id has access to.

Local exit routines have been added to PVM to prevent users from accessing nodes outside the scope of their authorization. PVM maintains an in-storage copy of the REMOTE AUTH file. Entries in this file define the valid destinations for the specified user IDs. A technique known as **generics** allows security personnel to control vast numbers of IDs and terminals with a one-line record.

PVM provides entry points for locally defined exits from the DVMNUE module. This module has been updated to call a new module, DVMSEC, which can be used to restrict users from accessing nodes outside their authority. Routines have been designed and implemented in DVMSEC:

- For the DVMNUE3 exit to restrict local users
- For the DVMNUE4 exit to restrict remote users
- For the DVMNUE5 exit to restrict commands
- To initialize the security exits table in storage from the REMOTE AUTH file
- To reinitialize this table from an updated REMOTE AUTH file by using a command, without shutting PVM down.

For more information, see "Developing Exits" on page 209.

© Copyright IBM Corp. 1991,1993

Installing and Building the Security Exits for PVM 2.1

To build a modified PVM module containing the security exits, you must have access to the base PVM text staging disk (49E), the PVM service staging disk (29E), and the security exit disk (39E). The following files are new or have changes to provide the security exits:

DVMSEC TXTSEC New security exits text deck

DVMNUE TXTSEC PVM Version 2 calling local, remote, command checks

DVMNME TXTSEC PVM Version 2 calling security initialization

DVMETA TXTSEC PVM Version 2 calling SEC RELOAD command processing

PVMSEC CNTRL Used for rebuilding the PVM component module.

Edit the PVM EXEC load list and add the following line under the DVMNME entry:

```
&1 &2 DVMSEC
```

Build the PVM MODULE using the PVMBLD command:

```
pvmbld pvm pvmsec
```

For more information on the PVMBLD command, see "PVMBLD" on page 315.

The REMOTE AUTH File

The REMOTE AUTH file contains fixed-length, 80-character records with one record per line. Input must be in uppercase alphanumeric characters. At least one blank must exist between each field of a record. Comments may appear after the last parameter, and any entry not preceded with an opcode will be considered a comment. Opcodes must appear in column 1. Valid opcodes are:

GATELINK Defines adjacent links to a node

REM-CMD Restricts users from issuing commands at local or remote nodes if the target is not the origin node

REM+USER Allows a user to continue past a remote node

SNA-USER Restricts local users with SNA terminals

VM-USER Restricts local users with non-SNA terminals.

In general, when the first match is found for a given set of conditions, the security exits allow the session to continue even if the next record disallows access. Therefore, the REMOTE AUTH table must be coded carefully. The exception to this rule is REM-CMD which will work just the opposite.

Note: User sessions and commands are not allowed if a REMOTE AUTH file is not found.

Note: All **unauthorized** attempts to access PVM are recorded in the SECAUDIT LOG file. All **valid** attempts are recorded by PVM in the PVM audit file if AUDIT is specified in the PVM CONFIG file. For examples of the SECAUDIT file see pages 452 and 455.

Using Generics

To minimize the size of the REMOTE AUTH file (and the search time required at runtime), a technique known as **generics** can be used in each opcode parameter. Use an asterisk * to indicate that any valid character may appear in the indicated position. For example, if you are specifying NYC*** as a node name, the following are valid because they all will match a user's node: NYCVM1, NYCDA, NYCZ, and NYC.

However, node NYCPUBS would not match as a user's node because there are more than three characters after NYC.

For an example of records using generics, see page 447

Reloading the REMOTE AUTH File

The REMOTE AUTH file can be reloaded without shutting PVM down. To do this, first update the REMOTE AUTH file. Then, the file can be reloaded into PVM by issuing the following command to PVM:

```
SEC RELOAD
```

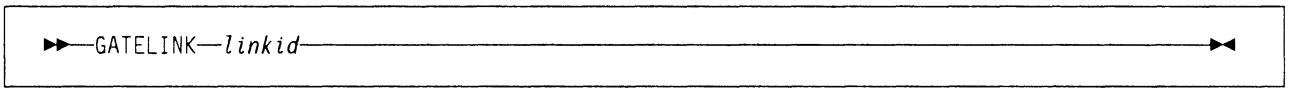
This is a privileged command, so you must be authorized to execute it.

Note: The disk might need to be accessed again to read the updated file.

Definitions of the REMOTE AUTH File Entries

The following section explains REMOTE AUTH records introduced with this support.

GATELINK



Purpose

If your installation wants to apply restrictions to remote sessions and commands across adjacent PVM links, use the GATELINK record.

Operands

linkid
is the node adjacent to the node where the GATELINK record resides.

Examples

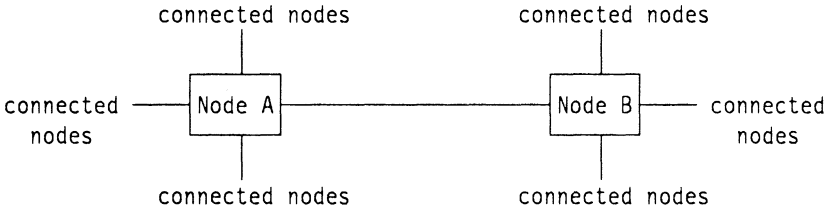
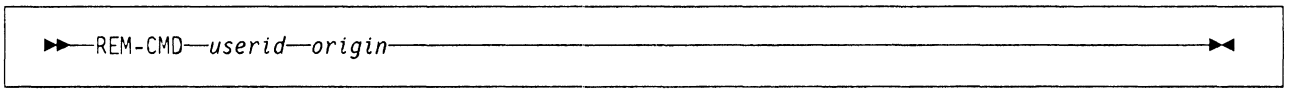


Figure 43. Example of Adjacent Nodes using the GATELINK Record

The REMOTE AUTH file at Node B contains a `GATELINK NODEA` record.

Node A is an adjacent node to Node B; they are connected by a PVM link. No commands will be allowed to or from Node A or nodes which use the link between Node A and Node B. All remote users in the network must have a REM+USER record to access any node which uses the link between Node A and Node B.

REM-CMD



Purpose

Use REM-CMD to restrict users from issuing commands at a specified local or remote node. Use this record if an adjacent node has not been defined by the GATELINK record.

Operands

userid
is the virtual machine identification of the local or remote user to restrict from issuing commands into the PVM network.

origin
are the nodes from which commands from *userid* are restricted.

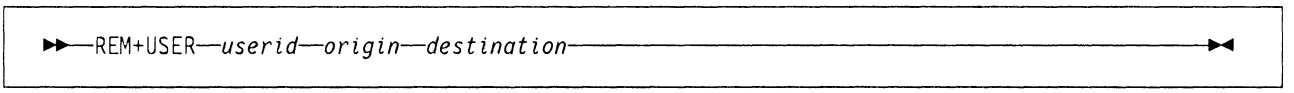
Note: This exit is not applicable if the target node ID for executing the command is also the origin node ID.

Examples

Below are two examples showing REM-CMD restrictions:

Command	Meaning
REM-CMD LES NYC***	• Restricts user LES from issuing commands at any node of the form NYC***. • User LES may issue commands at any node other than of the form NYC***. • All other users may issue commands at any node.
REM-CMD LES ********	• Restrict user LES from issuing commands at any node. • All other users may issue commands at any node.

REM+USER



Purpose

Use the REM+USER record when you need to identify users who are authorized to have sessions across a PVM link that has been defined with a GATELINK record. Unauthorized users will be denied access to the network.

Operands

userid
are the remote users allowed to continue to the next adjacent node or to logon to this node.

The session identification is in the form of:

- GRAF*xxx* for PVM dialed users
- The user's user ID for PASSTHRU, MPVM and IUCV users
- *luname* if the session origin is an SNA application
- CL*cc*-P*pp* for a remote cluster user attached to a PVM node
- Real address of a display station that is locally attached to a PVM node.

origin
are the starting nodes available to remote users.

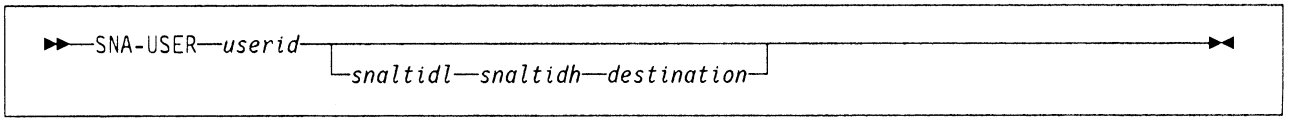
destination
are the target nodes available to remote users.

Examples

The following examples assume that a GATELINK has been defined.

Command	Meaning
`REM+USER TOM LOS*** NYCVM1`	• Permits user TOM from nodes of the form LOS*** access to node NYCVM1. • User TOM may not access any nodes other than NYCVM1, nor access NYCVM1 from nodes other than of the form LOS***. • All other users will be not be allowed access to any nodes because of the explicit GATELINK record.
`REM+USER TOM ******** NYCVM1`	• Permits user TOM from any node access to node NYCVM1. • All other users will not be allowed access to any nodes because of the explicit GATELINK record.

SNA-USER



Purpose

Use the SNA-USER record to restrict local users with SNA terminals. Local users can be restricted to a specific terminal address range, and/or restricted to specific destination nodes

The exit code differentiates an SNA user from a non-SNA user through the users' real terminal address supplied to it by PVM. SNA is assumed if the address is greater than five characters, or the first character is not an *r* or *l*.

Disconnected MPVM or IUCV users can be restricted with the SNA-USER record with a terminal ID of DSC.

Operands

userid
are the local users to be restricted.

The session identification is in the form of:

- GRAF*xxx* for PVM dialed users
- The user's user ID for PASSTHRU, MPVM, and IUCV users

snaltidl
is the SNA real terminal lower address boundary available to the local users being restricted. It is alphanumeric and up to eight characters long.

snaltidh
is the SNA real terminal higher address boundary available to the local users being restricted. It is alphanumeric and up to eight characters long.

destination
are the target nodes available to the local users being restricted.

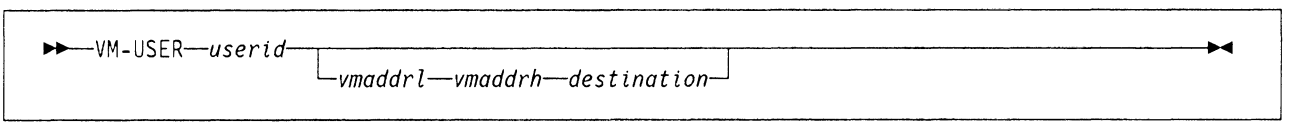
Examples

These are examples of the restrictions that SNA-USER places on users:

Command	Meaning
SNA-USER JIM H436**** H446**** NYC*****	• Restricts local user ID JIM on an SNA terminal to address range H436AAAA-H4469999 and access to target nodes of the form NYC***** only. • All other SNA terminal attached users may access any target node.

Command	Meaning
SNA-USER ******** H123**** H124**** NYCVM1	• Restricts all SNA terminal attached users to address range H123AAAA-H1249999 and access to target node NYCVM1 only. • Any SNA terminal attached users outside address range H123AAAA-H1249999 will not be allowed access to any target nodes.
SNA-USER MAINT DSC DSC NYCVM2	• Restricts MAINT user ID running a disconnected application accessed to target node NYCVM2 only. • All other SNA terminal attached users may access any target node.
SNA-USER VEN**** AAAAAAAA 99999999 CHICAGO	• Restricts all SNA terminal attached user IDs of the from VEN**** access to target node CHICAGO only. • All other SNA terminal attached users may access any target node as long as the VM user ID does not start with VEN.
SNA-USER ********	• Denies all SNA terminal attached users access to any node on the network no matter what address they are logged on to. **Note:** – This is the only time the parameters *snaltidl*, *snaltidh*, and *destination* are optional. They are optional because by omitting one or more of the parameters the example above is accomplished. All other times they are required. – This can be used to disallow all SNA terminal attached users access to the SEC PVM virtual machine.

VM-USER



Purpose

Use the VM-USER record to restrict local users with non-SNA terminals. Local users can be restricted to a specific terminal address range, and/or restricted to specific destination nodes.

Operands

userid
are the local users to be restricted.

The session identification is in the form of:

- GRAF*xxx* for PVM dialed users
- The user's user ID for PASSTHRU, MPVM, and IUCV users
- CL*cc*-P*pp* for a remote cluster user attached to the local PVM node
- Real address of a display station that is locally attached to the local PVM node.

vmaddrl
is the non-SNA real terminal lower address boundary available to the local users being restricted in the format of *hhh* or *hhhh* for local devices, L*hhh* or L*hhhh* for logical devices, and R*hhh* or R*hhhh* for CP remote cluster devices. The *h* is any hexadecimal character. The use of generics is not allowed with the *vmaddrl* parameter.

vmaddrh
is the non-SNA real terminal higher address boundary available to the local users being restricted in the format of *hhh* or *hhhh* for local devices, L*hhh* or L*hhhh* for logical devices, and R*hhh* or R*hhhh* for CP remote cluster devices. The *h* is any hexadecimal character. The use of generics is not allowed with the *vmaddrh* parameter.

destination
are the target nodes available to the local users being restricted.

Examples

These are examples of restrictions that VM-USER places on users:

Command	Meaning
`VM-USER DAVID  100 1FF VNETCMC`	• Restricts local user ID DAVID on a non-SNA terminal to address range 100-1FF and access to target node VNETCMC only. • All other non-SNA terminal attached users may access any target node.
`VM-USER ******** 100 1FF **VM1`	• Restricts all non-SNA terminal attached users to address range 100-1FF and access to target nodes of the form **VM1 only. • Any non-SNA terminal attached users outside address range 100-1FF will not be allowed access to any target nodes.
`VM-USER GRAF*** 000 FFF NYC***`	• Restricts all non-SNA terminal attached users having used the DIAL command access to target nodes of the form NYC*** only. • All other local non-SNA terminal attached PVM users will be allowed access to any target node as long as the VM user ID does not start with GRAF.

Command	Meaning
VM-USER ********	• Denies all non-SNA terminal attached users access to any node on the network no matter what address they are logged on to. **Note:** – This is the only time the parameters *vmaddrl*, *vmaddrh*, and *destination* are optional. They are optional because by omitting one or more of the parameters the example above is accomplished. All other times they are required. – This can be used to disallow all non-SNA terminal attached users access to the SEC PVM virtual machine.

Messages

Messages Returned to the User:

The sample security exits return the following messages to the user:

nodeid **SEC01: NOT ALLOWED TO DESTINATION**

Explanation: The user is not authorized to access the desired destination.

nodeid **SEC04: NOT ALLOWED TO ISSUE COMMAND**

Explanation: Indicates that the user/origin combination is not authorized to issue PVM commands at this node.

nodeid **SEC06: NO REMOTE AUTH TABLE FOUND**

Explanation: Indicates that a REMOTE AUTH table could not be found on any disk accessed by PVM's user ID. Thus, no user sessions or commands will be allowed.

Usage Notes

1. The node ID at the beginning of the messages above is the node ID of the PVM machine that issued that message.
2. No syntax error messages are produced for invalid records in the REMOTE AUTH file.

Messages Returned to the PVM Console

The sample security exits return the following messages to the PVM console:

INVALID SUBCOMMAND ON SEC COMMAND

Explanation: The SEC command had a subcommand other than RELOAD.

COULD NOT LOCATE REMOTE AUTH FILE

Explanation: The REMOTE AUTH file is not on any disks accessed by the PVM user ID.

COULD NOT ALLOCATE STORAGE FOR REMOTE AUTH TABLES

Explanation: Not enough storage is available for loading the in-storage tables of the REMOTE AUTH file.

SECAUDIT LOG FILE CANNOT BE WRITTEN

Explanation: An error occurred while attempting to write an entry to the SECAUDIT LOG file.

Usage Notes

1. Messages are sent to the PVM user ID using the CMS macro LINEWRT.
2. Use of LINEWRT (and other CMS functions) should be limited to avoid possible conflicts between CMS and the PVM operating environment.

Format of SECAUDIT LOG Records

Invalid session or command execution attempts are recorded in the SECAUDIT LOG file on the SEC PVM's filemode A minidisk. SECAUDIT is a fixed-length file created and updated by DVMSEC. SECAUDIT LOG contains these types of records which are created for the following situations, unsuccessful attempts:

- By a user to gain access to the PVM network through a local system (DVMNUE3 entry point).
- To connect to or through a remote PVM node (DVMNUE4 entry point).
- To issue commands to or through a remote PVM node or from a user ID which has been excluded from issuing commands in the PVM network (DVMNUE5 entry point).

You should review the SECAUDIT LOG file periodically because it will continue to grow as unauthorized attempts are added to it.

Unauthorized attempts are all recorded in the SECAUDIT LOG file. A 3, 4, or 5 is written to the SECAUDIT LOG after the DATE/TIME indicating the type of attempt, where the attempt originated, and the destination node attempted in the case of sessions. In the case of commands, where the command originated and what command execution was attempted appears on the lines following the 5.

Table 31. SECAUDIT Record Formats

Record Format Type	Column	Contents
3 and 4	1	Date
	10	Time
	19	SEC record format type 3 or 4
	21	Origin node ID
	30	Origin user ID
	39	Terminal address
	48	Destination node ID (attempted)
		Note: The terminal address will be blank for unauthorized access attempts by an IUCV user at a remote node. The terminal address will be DSC for unauthorized access attempts by an MPVM or IUCV user at the local node.
5	1	Date
	10	Time
	19	SEC record format type 5
	21	Origin node ID of user issuing command
	30	Origin user ID of user issuing command
	39	Node ID where command was to be executed
	48	Command to be executed (up to column 80)

See pages 452 and 455 for more examples of a SECAUDIT LOG file.

Controlling Local Users Using the REMOTE AUTH Table

The following REMOTE AUTH table controls the access to PVM for local users on the CAMBRIDG system.

```
VM-USER GATEDOC   0404  040F PARISVM
VM-USER GUEST*    0410  0410 NYCVMX
VM-USER GUEST**   0400  0410 NYCVMT
VM-USER GUEST4    0000  1FFF CAMBVM2
VM-USER XXXPAR*   0000  1FFF
REM-CMD XXXPAR*   CAMBRIDG
REM-CMD GUEST*    CAMBRIDG
```

Figure 44. REMOTE AUTH Table for Local Users

User ID	Access
GATEDOC	Can only access node PARISVM from a terminal with an address between 0404 and 040F.
GUEST* GUEST**	Represents user IDs GUESTA through GUESTZ, and user IDs GUEST0 through GUEST9. The two asterisks ** of user ID GUEST** can be any combination of A-Z and 0-9. User ID GUEST* and GUEST** can access NYCVMT from a terminal on addresses 400 through 410. User ID GUEST* can also access NYCVMX from a terminal on address 410. **Note:** The above list represents all the nodes that GUEST** and GUEST* can access and from which address.

User ID	Access
GUEST4	• Can access node CAMBVM2 from any address between 000 and 1FFF. • Can access NYCVMT from a terminal on address between 400 through 410. • Can access NYCVMX from a terminal on address 410. **Note:** The above list represents all the nodes that GUEST4 can access and from which address.
XXXPAR*	Represents user IDs XXXPARA through XXXPARZ, and user IDs XXXPAR0 through XXXPAR9. None of these user IDs can access PVM because no destination node ID is specified. This indicates that this user ID is not to have access to PVM no matter what address is specified on the REMOTE AUTH record. Since no destination was specified, the terminal address range has no effect because there will always be a mismatch on the destination. **Note:** User IDs XXXPAR* and GUEST* Are unable to execute PVM commands on remote nodes.

DVMNUE3 Testing

The following are test cases using the REMOTE AUTH table controlling local users.

Table 32 (Page 1 of 2). REMOTE AUTH Table Controlling Local Users

User ID	Address	PASSTHRU Node or Command to PVM	Access
GATEDOC	40F	CAMBVM2	Disallowed
		NYCVMX	Disallowed
		NYCVMT	Disallowed
		PARISVM	Allowed
		SM PVM CMD PARIS Q SYS	Allowed
	410	CAMBVM2	Disallowed
		NYCVMX	Disallowed
		NYCVMT	Disallowed
		PARISVM	Disallowed
	82A	CAMBVM2	Disallowed
		NYCVMX	Disallowed
		NYCVMT	Disallowed
		PARISVM	Disallowed
GUEST3	410	NYCVMX	Allowed
		NYCVMT	Allowed
		CAMBVM2	Disallowed
		PARISVM	Disallowed
		SM PVM CMD LOSVMV Q SYS	Disallowed
	82A	CAMBVM2	Disallowed
		NYCVMX	Disallowed
		NYCVMT	Disallowed
		PARISVM	Disallowed
		SM PVM CMD LONVM0 Q SYS	Disallowed

Table 32 (Page 2 of 2). REMOTE AUTH Table Controlling Local Users

User ID	Address	PASSTHRU Node or Command to PVM	Access
GUEST4	40F	CAMBVM2	Allowed
		CAMBVM3	Disallowed
		NYCVMT	Allowed
		NYCVMX	Disallowed
		SM PVM CMD NYCVMV Q SYS	Disallowed
	410	NYCVMX	Allowed
		NYCVMT	Allowed
		NYCGATE	Disallowed
	82A	CAMBVM2	Allowed
		NYCVMT	Disallowed
		NYCVMX	Disallowed
XXXPAR3	410	PARISVM	Disallowed
		NYCGATE	Disallowed
		SM PVM CMD PARVMV Q SYS	Disallowed

SECAUDIT LOG

The following is the SECAUDIT LOG associated with the REMOTE AUTH table controlling local users.

```
09/21/91 06:35:24 3 CAMBRIDG GUEST4   040F     CAMBVM3
09/21/91 06:35:45 3 CAMBRIDG GUEST4   040F     NYCVMX
09/21/91 06:36:45 5 CAMBRIDG GUEST4   NYCVMV   CMD NYCVMV Q SYS
09/21/91 06:36:55 3 CAMBRIDG GUEST4   0410     NYCGATE
09/21/91 06:37:44 3 CAMBRIDG GUEST3   0410     CAMBVM2
09/21/91 06:37:50 3 CAMBRIDG GUEST3   0410     PARISVM
09/21/91 06:38:13 5 CAMBRIDG GUEST3   LOSVMV   CMD LOSVMV Q SYS
09/21/91 06:38:41 3 CAMBRIDG XXXPAR3  0410     PARISVM
09/21/91 06:38:47 3 CAMBRIDG XXXPAR3  0410     NYCGATE
09/21/91 06:38:51 5 CAMBRIDG XXXPAR3  PARVMV   CMD PARVMV Q SYS
09/21/91 06:40:30 3 CAMBRIDG GUEST4   082A     NYCVMT
09/21/91 06:40:34 3 CAMBRIDG GUEST4   082A     NYCVMX
09/21/91 06:40:52 3 CAMBRIDG GUEST3   082A     CAMBVM2
09/21/91 06:40:58 3 CAMBRIDG GUEST3   082A     NYCVMX
09/21/91 06:41:02 3 CAMBRIDG GUEST3   082A     NYCVMT
09/21/91 06:41:06 3 CAMBRIDG GUEST3   082A     PARISVM
09/21/91 06:41:15 5 CAMBRIDG GUEST3   LONVM0   CMD LONVM0 Q SYS
09/21/91 06:41:32 3 CAMBRIDG GATEDOC  040F     CAMBVM2
09/21/91 06:41:39 3 CAMBRIDG GATEDOC  040F     NYCVMX
09/21/91 06:41:41 3 CAMBRIDG GATEDOC  040F     NYCVMT
09/21/91 06:41:45 3 CAMBRIDG GATEDOC  0410     CAMBVM2
09/21/91 06:42:52 3 CAMBRIDG GATEDOC  0410     NYCVMX
09/21/91 06:42:56 3 CAMBRIDG GATEDOC  0410     NYCVMT
09/21/91 06:42:02 3 CAMBRIDG GATEDOC  0410     PARISVM
09/21/91 06:42:25 3 CAMBRIDG GATEDOC  082A     CAMBVM2
09/21/91 06:42:32 3 CAMBRIDG GATEDOC  082A     NYCVMX
09/21/91 06:42:39 3 CAMBRIDG GATEDOC  082A     NYCVMT
09/21/91 06:43:19 3 CAMBRIDG GATEDOC  082A     PARISVM
```

Figure 45. SECAUDIT LOG for Local Users

Controlling Remote Users by Using the REMOTE AUTH Table

The following REMOTE AUTH table at a remote gateway node controls access between systems which use PVM links defined at this node. This machine is not the PVM machine that users normally access. Traffic through this node is destined to or from another location. Users are not allowed to access PVM locally. Commands cannot be sent on a link which goes through this gateway node.

```
VM-USER  ********
REM+USER DRBIM    BOSTON   CAMBRIDG
REM+USER DRBIM    BOSTON   NEWYORK
REM+USER DAVID*   CAMBRIDG BOSTON
REM+USER DAVID*   CAMBRIDG LONDON
REM+USER BILL     CAMBRIDG BOSTON
REM+USER JARRELL  CAMBRIDG LONDON
REM+USER GATEDOC  CAMBRIDG ******
REM+USER NODEADMN CAMBRIDG ********
REM-CMD  ******** GATEWAY
GATELINK BOSTON
GATELINK CAMBRIDG
```

Figure 46. REMOTE AUTH Table at a Remote Gateway Node

Figure 47 shows how the PVM network is connected:

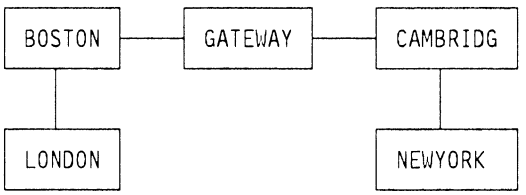


Figure 47. The PVM Network

The REMOTE AUTH table shown in Figure 46 disallows access from any local terminal to PVM by using the first VM-USER statement. Since no destination node ID is specified, no access is allowed by any local user. Also, no commands can be issued into the PVM network by any local user ID by using the REM-CMD statement.

Two gateway links are defined: CAMBRIDG and BOSTON. Any traffic routed between these two links is checked against the REMOTE AUTH table. Also, any commands routed between these two links cannot be executed.

User DRBIM on node BOSTON can only access the CAMBRIDG and NEWYORK nodes. No other user from BOSTON or LONDON can access the CAMBRIDG, NEWYORK, or GATEWAY nodes.

The following CAMBRIDG user IDs have access to node BOSTON:

- DAVID0 thru DAVID9
- DAVIDA thru DAVIDZ
- BILL
- GATEDOC
- NODEADMN.

The following CAMBRIDG user IDs are given access to node LONDON:

- DAVID0 thru DAVID9
- DAVIDA thru DAVIDZ
- GATEDOC
- JARRELL
- NODEADMN.

User ID NODEADNM can also access the GATEWAY node.

No other user from CAMBRIDG or NEWYORK can access the BOSTON, LONDON, or GATEWAY nodes.

DVMNUE4 Testing

The following are test cases using the REMOTE AUTH table controlling remote users.

Table 33. REMOTE AUTH Table Controlling Remote Users

From Node	From User ID	PASSTHRU Node	Access
CAMBRIDG	DAVID2A	BOSTON LONDON GATEWAY NEWYORK	Disallowed Disallowed Disallowed Allowed
CAMBRIDG	DAVID2 or GATEDOC	BOSTON LONDON GATEWAY NEWYORK	Allowed Allowed Disallowed Allowed
CAMBRIDG	BILL	BOSTON LONDON GATEWAY NEWYORK	Allowed Disallowed Disallowed Allowed
CAMBRIDG	JARRELL	BOSTON LONDON GATEWAY NEWYORK	Disallowed Allowed Disallowed Allowed
CAMBRIDG	NODEADM	BOSTON LONDON GATEWAY NEWYORK	Allowed Allowed Allowed Allowed
NEWYORK	OTHER	BOSTON LONDON GATEWAY CAMBRIDG	Disallowed Disallowed Disallowed Allowed
BOSTON	DRBIM	CAMBRIDG NEWYORK GATEWAY LONDON	Allowed Allowed Disallowed Allowed
LONDON	OTHER	CAMBRIDG NEWYORK GATEWAY BOSTON	Disallowed Disallowed Disallowed Allowed

SECAUDIT LOG

The following is the SECAUDIT LOG associated with the REMOTE AUTH table controlling remote users.

```
09/21/91 09:39:49 4 CAMBRIDG DAVID2A  082A BOSTON
09/21/91 09:40:51 4 CAMBRIDG DAVID2A  082A LONDON
09/21/91 09:41:22 4 CAMBRIDG DAVID2A  082A GATEWAY
09/21/91 09:41:30 4 CAMBRIDG GATEDOC  082A GATEWAY
09/21/91 09:41:35 4 CAMBRIDG BILL     082A LONDON
09/21/91 09:41:39 4 CAMBRIDG BILL     082A GATEWAY
09/21/91 09:41:35 4 CAMBRIDG JARRELL  082A BOSTON
09/21/91 09:41:39 4 CAMBRIDG JARRELL  082A GATEWAY
09/21/91 09:43:45 4 NEWYORK  OTHER    082A BOSTON
09/21/91 09:43:51 4 NEWYORK  OTHER    082A LONDON
09/21/91 09:43:56 4 NEWYORK  OTHER    082A GATEWAY
09/21/91 09:44:00 4 BOSTON   DRBIM    082A GATEWAY
09/21/91 09:44:14 4 LONDON   OTHER    082A CAMBRIDG
09/21/91 09:46:58 4 LONDON   OTHER    082A NEWYORK
09/21/91 09:47:16 4 LONDON   OTHER    082A GATEWAY
```

Figure 48. SECAUDIT LOG for Remote Users

DMTI end

Appendix K. CVIEW Product Panels

CVIEW displays menu and data-entry panels to its users during setup mode. In addition, consultants and moderators receive control-mode panels during CVIEW sessions by pressing the PA1 key. If your establishment uses the NOTIFY panel, CVIEW displays it to you at preset intervals during session mode. This appendix presents all the CVIEW product panels and indicates when they are displayed.

Setup-mode Panels

Setup mode begins after you enter 'DIAL CVIEW'. The first panel displayed is the ***HOME*** panel:

```
HOME              COOPERATIVE VIEWING FACILITY (CVIEW)
--------------------------------------------------------------------------------
 Enter an item number.

     Consult Session

         1  Receive assistance
         2  Give assistance

     Conference Session

         3  Attend a conference
         4  Moderate a conference




 Press ENTER after you enter an item number.
--------------------------------------------------------------------------------
PF:  1= Help     2=          3= Quit     4=          5=          6=
PF:  7=          8=          9=         10=         11=         12=

    ===>
```

Enter a number (1-4) to select your role in the CVIEW session. After you select your role, you see one of the following panels:

1. A **recipient** sees the ***RECEIVE*** panel.
2. A **consultant** sees the ***GIVE*** panel.
3. An **attendee** sees the ***ATTEND*** panel.
4. A **moderator** sees the ***MODERATE*** panel.

© Copyright IBM Corp. 1991,1993

Consult Session Setup (Recipient)

```
RECEIVE                      CVIEW - CONSULTANT SELECTION
--------------------------------------------------------------------------------
 Contact a consultant by telephone before entering an item number.

    1    Consultant not listed
    2    Consultant #1           Product 1                    555-0001
    3    Consultant #2           Product 2                    555-0002
    4    Consultant #3           Product 3                    555-0003
    5    Consultant #4           Product 4                    555-0004
    6    Consultant #5           Product 5                    555-0005
    7    Consultant #6           Product 6                    555-0006
    8    Consultant #7           Product 7                    555-0007
    9    Consultant #8           Product 8                    555-0008
   10    Consultant #9           Product 9                    555-0009
   11    Consultant #10          Product 10                   555-0010
   12    Consultant #11          Product 11                   555-0011


 Press ENTER after you enter an item number.
--------------------------------------------------------------------------------
PF:  1= Help     2=          3= Quit     4= Home     5=          6=
PF:  7=          8=          9=         10=         11=         12=

    ===>
```

The ***RECEIVE*** panel appears after the HOME panel when you are setting up a consult session as a recipient. Enter a number from 1 - 12 to select a consultant.

```
IDRECV            CVIEW - UNLISTED CONSULTANT SELECTION
--------------------------------------------------------------------------------
 Enter the identification provided by the consultant.

     Consultant's identification:  ===>          1 to 8 characters (no blanks)

 The consultant identification you enter must also be entered by the
 consultant.  This identification allows CVIEW to associate your
 terminal with the consultant's terminal in a consult session.









 Press ENTER after you enter the identification.
--------------------------------------------------------------------------------
PF:  1= Help     2=          3= Quit     4= Home     5=          6=
PF:  7=          8=          9=         10=         11=         12=

    ===>
```

CVIEW displays the ***IDRECV*** (ID RECEIVE) panel if you entered 1 to select a consultant who is not already listed with CVIEW. Now, you must enter the consultant's identification as he gives it to you.

```
RECVWAIT                    CVIEW - RECIPIENT WAIT
--------------------------------------------------------------------------------
 You and the consultant should be in contact by telephone.  You are waiting
 for the consultant to establish a consult session with you.

 You selected to receive assistance from consultant #XX: CONSULTANTID.
 The phone number for this consultant is XXXXXXXXXXXXXX.

The consultant must do the following to establish a consult session with you:

  1) Start from the VM system logo (where CVIEW is running)
  2) Clear the screen and issue the command: DIAL CVIEW
  3) Select option 2 from the Home panel
  4) Specify the same consultant identification you selected (see above)

  5) The VM system logo will be displayed when the consult session starts

 If you do not wish to wait for the consultant, press the Home PF key to
 return to the Home panel.

 No further action is required from you.
--------------------------------------------------------------------------------
PF:  1= Help     2=          3= Quit     4= Home     5=          6=
PF:  7=          8=          9=         10=         11=         12=

    ===>
```

CVIEW displays the ***RECVWAIT*** (RECEIVE WAIT) panel when you have responded to all the recipient setup panels. You are waiting for the consultant to complete his setup panels. The logo for your VM/ESA system will appear when the consultant completes his setup panels.

Consult Session Setup (Consultant)

```
GIVE                    CVIEW - CONSULTANT IDENTIFICATION
------------------------------------------------------------------------------
 Enter an item number.  The recipient must enter the same item number.

    1    Consultant not listed
    2    Consultant #1            Product 1                    555-0001
    3    Consultant #2            Product 2                    555-0002
    4    Consultant #3            Product 3                    555-0003
    5    Consultant #4            Product 4                    555-0004
    6    Consultant #5            Product 5                    555-0005
    7    Consultant #6            Product 6                    555-0006
    8    Consultant #7            Product 7                    555-0007
    9    Consultant #8            Product 8                    555-0008
   10    Consultant #9            Product 9                    555-0009
   11    Consultant #10           Product 10                   555-0010
   12    Consultant #11           Product 11                   555-0011


 Press ENTER after you enter an item number.
------------------------------------------------------------------------------
PF:  1= Help     2=          3= Quit     4= Home     5=          6=
PF:  7=          8=          9=         10=         11=         12=

    ===>
```

The ***GIVE*** panel appears after the HOME panel when you are setting up a consult session as a consultant. Enter a number from 1 - 12 to select your consultant identification.

```
IDGIVE             CVIEW - UNLISTED CONSULTANT IDENTIFICATION
------------------------------------------------------------------------------
 Enter the data.

     Consultant identification: ===>          1 to 8 characters (no blanks)

     Display input PF key:      ===>          PF1,PF2....,PF24 or NONE
     (for use during consult session)

  The consultant identification you specify must also be entered by the
  recipient.  This identification allows CVIEW to associate your
  terminal with the recipient's terminal in a consult session.

  You may press the PA1 key during the consult session to display the
  Consult Session Control panel.



  Press ENTER after you enter the data.
------------------------------------------------------------------------------
 PF:  1= Help     2=          3= Quit     4= Home     5=          6=
 PF:  7=          8=          9=         10=         11=         12=

     ===>
```

CVIEW displays the ***IDGIVE*** (ID GIVE) panel if you entered 1 to indicate that you are an unlisted consultant. You must enter your identification and tell it to the recipient. You can use this panel to define a PF key to display input before it is processed.

```
CONSPASS              CVIEW - CONSULTANT PASSWORD
-------------------------------------------------------------------------------
 Enter the data.

     Consultant password:       ===>          1 to 8 characters (no blanks)

     Display input PF key:      ===>          PF1,PF2....,PF24 or NONE
     (for use during consult session)

 You may press the PA1 key during the consult session to display the
 Consult Session Control panel.




 Press ENTER after you enter the data.
-------------------------------------------------------------------------------
PF:  1= Help     2=          3= Quit     4= Home     5=          6=
PF:  7=          8=          9=         10=         11=         12=

    ===>
```

CVIEW displays the ***CONSPASS*** (CONSULTANT PASSWORD) panel if you are a listed consultant with a password. Enter your password. You can also use this panel to define a PF key to display input before it is processed.

```
REDISP                        CVIEW - DISPLAY INPUT
-------------------------------------------------------------------------------
 Enter the data.

     Display Input PF key:    ===>              PF1,PF2,...,PF24 or NONE
     (for use during consult session)

 You may press the PA1 key during the consult session to display the
 Consult Session Control panel.
    .




 Press ENTER after you enter the data.
-------------------------------------------------------------------------------
PF:  1= Help     2=          3= Quit     4= Home     5=          6=
PF:  7=          8=          9=         10=         11=         12=

    ===>
```

CVIEW displays the ***REDISP*** (RE-DISPLAY) panel if you are a listed consultant without a password. You can use this panel to define a PF key to display input before it is processed.

```
GIVEWAIT                    CVIEW - CONSULTANT WAIT
------------------------------------------------------------------------------
 You and the recipient should be in contact by telephone.  You are waiting
 for the recipient to establish a consult session with you.

 You specified your consultant identification as #XX: CONSULTANTID.

 The recipient must do the following to establish a consult session with you.
   1) Start from the VM system logo (where CVIEW is running)
   2) Clear the screen and issue the command: DIAL CVIEW
   3) Select option 1 from the Home panel
   4) Select the same consultant identification you specified (see above)
   5) The VM system logo will be displayed when the consult session starts

 If you do not wish to wait for the recipient, press the Home PF key
 to return to the Home panel.


 No further action is required from you.
------------------------------------------------------------------------------
PF:  1= Help     2=          3= Quit     4= Home     5=          6=
PF:  7=          8=          9=         10=         11=         12=

    ===>
```

CVIEW displays the ***GIVEWAIT*** panel when you have responded to all the consultant setup panels. You are waiting for the recipient to complete his setup panels. The logo for your VM/ESA system will appear when the recipient completes his setup panels.

Conference Session Setup (Attendee)

```
ATTEND             CVIEW - CONFERENCE PARTICIPANT IDENTIFICATION
------------------------------------------------------------------------------
 Enter the data.

     Conference name:    ===>                 1 to 8 characters (no blanks)

     Your Name:          ===>                 1 to 16 characters

 If you wish to leave the conference after it has started, you may press
 the PA1 key.  If the moderator has assigned you to be a presenter in the
 conference, the PA1 key will perform the function normally associated
 with it.




 Press ENTER after you enter the data.
------------------------------------------------------------------------------
PF:  1= Help     2=          3= Quit     4= Home     5=          6=
PF:  7=          8=          9=         10=         11=         12=

    ===>
```

The ***ATTEND*** panel appears after the HOME panel when you are setting up a conference session as an attendee. Enter the conference name provided by the moderator, and enter your own name.

```
PASSWORD                  CVIEW - CONFERENCE PASSWORD
------------------------------------------------------------------------------
 Enter the password for the conference.

     Conference password:  ===>                1 to 8 characters (no blanks)









 Press ENTER after you enter the password.
------------------------------------------------------------------------------
PF:  1= Help     2=          3= Quit     4= Home     5=          6=
PF:  7=          8=          9=         10=         11=         12=

    ===>
```

CVIEW displays the ***PASSWORD*** panel if the moderator assigns a password to the conference. Enter the conference password now.

```
CONFWAIT                  CVIEW - CONFERENCE WAIT
------------------------------------------------------------------------------
 A VM system logo will be displayed when the conference starts.

 You are waiting for conference XXXXXXXX to start.

 If you wish to leave the conference after it has started, you may press
 the PA1 key.  If the moderator has assigned you to be a presenter in the
 conference, the PA1 key will perform the function normally associated
 with it.

 A Notify panel may be displayed occasionally by CVIEW.  In order for the
 conference to continue, you must press ENTER when the Notify panel is
 displayed.

 If a conference password is required, you will be prompted for the password
 on the Conference Password panel.

 No further action is required from you.
------------------------------------------------------------------------------
PF:  1= Help     2=          3= Quit     4= Home     5=          6=
PF:  7=          8=          9=         10=         11=         12=

    ===>
```

CVIEW displays the ***CONFWAIT*** (CONFERENCE WAIT) panel after you complete all the setup panels to attend a conference. You are waiting for the moderator to start the conference. The logo for your VM/ESA system will appear when the moderator starts the conference.

Conference Session Setup (Moderator)

```
MODERATE              CVIEW - MODERATOR IDENTIFICATION
--------------------------------------------------------------------------------
 Enter the data.

     Conference Name:      ===>                1 to 8 characters (no blanks)

     Your Name:            ===>                1 to 16 characters

     Display Input PF key: ===>                PF1,PF2,...,PF24 or NONE
     (for use during conference)

     Conference Password:  ===>                1 to 8 characters (no blanks)
     (optional)


 You may press the PA1 key during the conference to display the Conference
 Control panel.

 Press ENTER after you enter the data.
--------------------------------------------------------------------------------
PF:  1= Help     2=          3= Quit     4= Home     5=          6=
PF:  7=          8=          9=         10=         11=         12=

    ===>
```

Enter the conference name on this panel and tell it to the attendees. Enter your name also. You can use this panel to define a PF key to display input before it is processed. In addition, you can assign a password to the conference. If you do so, tell the password to the attendees.

```
XXXXXXXX              CVIEW - CONFERENCE PARTICIPANTS
--------------------------------------------------------------------------------
 You may change a participant's status by entering one of the following
 item numbers next to the participant's name and pressing ENTER:

     1  Allow a participant to present in the conference (presenter)
     2  Allow a participant to only observe the conference (attendee)
     3  Drop a participant from the conference

         Participant      Status               Participant      Status
         XXXXXXXXXXXXXXXX Moderator     ===>
 ===>                                   ===>
 ===>                                   ===>
 ===>                                   ===>
 ===>                                   ===>
 ===>                                   ===>

 Press ENTER if you enter an item number next to a participant's name.
 Press the SESSION PF key to start the conference session.
--------------------------------------------------------------------------------
PF:  1= Help     2=          3= Quit     4= Home     5=          6=
PF:  7=          8=          9=         10=         11=         12= Session

    ===>
```

The name you entered for the conference appears in the upper left-hand corner of this panel. You can now assign presenter status to any of the attendees or drop attendees from the conference. When you are ready to start the conference

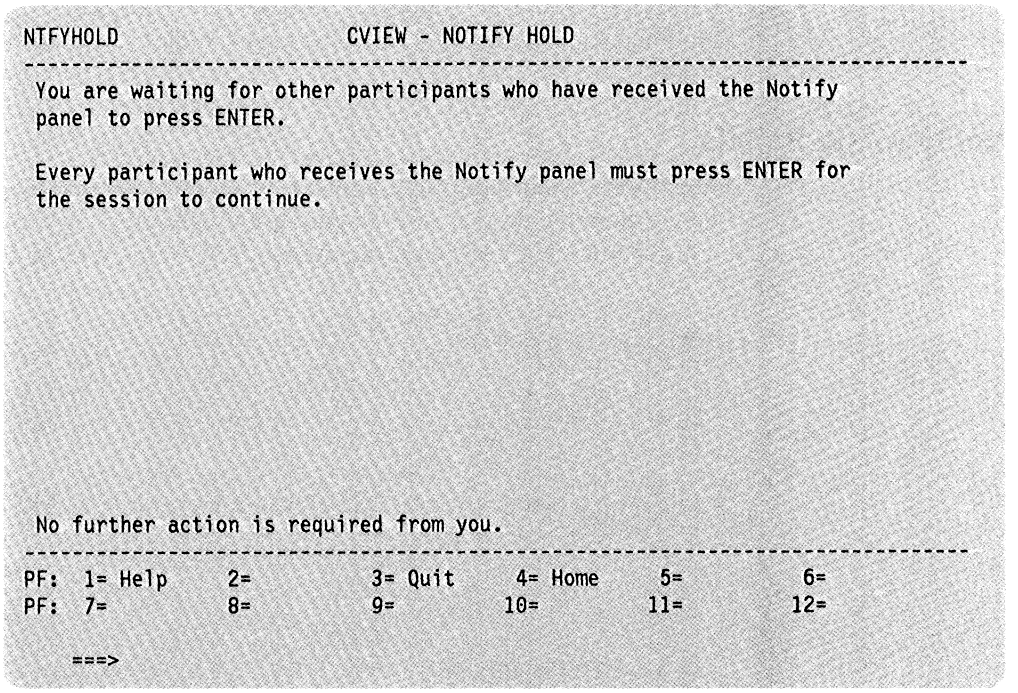
session, press the PF key for SESSION. After you press the SESSION PF key, the logo for your VM/ESA system will appear, indicating that the session has begun.

Notify-mode Panels

Notify mode occurs during a consult or conference session if the use of the NOTIFY panel was selected for your establishment. The session cannot continue until every CVIEW user who receives a NOTIFY panel has responded to it.

```
NOTIFY                          CVIEW - NOTIFY
------------------------------------------------------------------------------
 This Notify panel is being displayed by the Cooperative Viewing Facility
 (CVIEW) to make you aware you are running under the control of
 CVIEW.

 You are currently in a CVIEW session, someone else is watching the same
 screen images you are displaying.

 If you press the QUIT or HOME PF key while this panel is displayed, the
 session will be terminated.

 If you were not aware you were using this facility, contact your Computer
 Center Management.

 Every participant who receives a Notify panel must press ENTER for the
 session to continue

 Press ENTER to continue.
------------------------------------------------------------------------------
PF:  1= Help     2=          3= Quit     4= Home     5=          6=
PF:  7=          8=          9=         10=         11=         12=

    ===>
```

CVIEW displays the ***NOTIFY*** panel at preset intervals during consult and conference sessions. Press ENTER to continue the CVIEW session. If you do not want to continue the session, press the PF key assigned to either HOME or QUIT.



CVIEW displays the ***NTFYHOLD*** (NOTIFY HOLD) panel after you press ENTER in response to the NOTIFY panel. The NTFYHOLD panel indicates that other session participants have not yet pressed ENTER in response to their NOTIFY panels.

Control-mode Panels

A consultant or moderator enters control mode during a CVIEW session by pressing the PA1 key to access a control panel. During control mode, the consult or conference session is suspended.

Consultant Control

```
CONSCTL              CVIEW - CONSULT SESSION CONTROL
--------------------------------------------------------------------------------
 You may change the data below by entering a new value.

 Currently, your Display Input PF key is set to XXXX.

     Display Input PF key:      ===>              PF1,PF2,...,PF24 or NONE
     (for use during consult session)

 If you had intended the PA1 key to perform the function normally
 associated with it, press the PA1 key again now.

 The consult session cannot continue while this panel is being displayed.




 Press ENTER to continue.
--------------------------------------------------------------------------------
PF:  1= Help     2=          3= Quit     4= Home     5=          6=
PF:  7=          8=          9=         10=         11=         12=

    ===>
```

After you, as a consultant, press the PA1 key, CVIEW displays the ***CONSCTL*** (CONSULT SESSION CONTROL) panel. You can now change the PF key assignment for input display. In addition, you can press PA1 again while this panel is on the screen and the host or application program will perform the function normally associated with the PA1 key. The consult session continues when you press ENTER or PA1 in response to this panel.

Moderator Control

```
CONFCTL                    CVIEW - CONFERENCE CONTROL
--------------------------------------------------------------------------------
 You may change the data below by entering a new value.

 Currently, your Display Input PF key is set to XXXX.

     Display Input PF key:    ===>              PF1,PF2,...,PF24 or NONE
     (for use during conference)

     Display the Conference
     Participants panel:      ===>              Yes or No


 If you had intended the PA1 key to perform the function normally
 associated with it, press the PA1 key again now.

 The conference cannot continue while this panel is being displayed.

 Press ENTER to continue.
--------------------------------------------------------------------------------
PF:  1= Help     2=          3= Quit     4= Home     5=          6=
PF:  7=          8=          9=         10=         11=         12=

    ===>
```

After you, as a moderator, press the PA1 key, CVIEW displays the ***CONFCTL*** (CONFERENCE CONTROL) panel. You can now change the PF key assignment for input display or request the CONFERENCE PARTICIPANTS panel. In addition, you can press PA1 again and the host or application program will perform the function normally associated with the PA1 key.

```
XXXXXXXX              CVIEW - CONFERENCE PARTICIPANTS
--------------------------------------------------------------------------------
 You may change a participant's status by entering one of the following
 item numbers next to the participant's name and pressing ENTER:

     1  Allow a participant to present in the conference (presenter)
     2  Allow a participant to only observe the conference (attendee)
     3  Drop a participant from the conference

         Participant      Status               Participant      Status
         XXXXXXXXXXXXXXXX Moderator     ===>
 ===>                                   ===>
 ===>                                   ===>
 ===>                                   ===>
 ===>                                   ===>
 ===>                                   ===>

 Press ENTER if you enter an item number next to a participant's name.
 Press the SESSION PF key to return to the conference session.
--------------------------------------------------------------------------------
PF:  1= Help     2=          3= Quit     4= Home     5=          6=
PF:  7=          8=          9=         10=         11=         12= Session

    ===>
```

CVIEW displays the ***CONFERENCE PARTICIPANTS*** panel when you request it from the CONFERENCE CONTROL panel. The conference session can resume while this panel is displayed, if a presenter exists. From this panel, you can

change a participant's status to attendee or presenter, and drop attendees from the conference session. Type the appropriate number by a participant's name and press ENTER. You return to the conference in progress by pressing the SESSION PF key.

Glossary

This section explains or defines selected terms, acronyms, and abbreviations that appear in this book.

A

abend. (1) Abnormal end of task. (2) Synonym for *abnormal termination*.

abend dump. The contents of main storage, or part of main storage, written to an external medium for debugging an error condition that resulted in the termination of a task before its regular completion.

abnormal end of task (abend). Termination of a task before its completion because of an error condition that cannot be resolved by recovery facilities while the task is executing.

abnormal termination. The ending of processing before planned termination. Synonymous with *abend*.

accept. (1) Allowing a connection to the user's virtual machine from another virtual machine or from the user's own virtual machine. (2) For a VTAM application program, to establish a session with an LU in response to a CINIT request from an SSCP. The session-initiation request may begin when a terminal user logs on, a VTAM application program issues a macroinstruction, or a VTAM operator issues a command. See also *acquire*.

access method. A technique for moving data between main storage and I/O devices.

access method services (AMS). The utility that allocates and configures VSAM data sets. AMS is invoked through the AMSERV command in CMS.

access security. Information that a target LU and target transaction program use to verify whether a source program is authorized to make a connection. This information consists of a user ID and, possibly, a password.

ACF. Advanced communications function.

ACF/VTAM. Advanced Communications Function for Virtual Telecommunications Access Method.

acquire. (1) For a VTAM application program, to initiate and establish a session with another LU. The acquire process begins when the application program issues a macroinstruction. See also *accept*. (2) To take over resources that were formerly controlled by an access method in another domain, or to resume control of resources that were controlled by this domain but released. Contrast with *release*. See also *resource takeover*.

activate. To make a resource of a node ready to perform the functions for which it was designed. Contrast with *deactivate*.

active. (1) The state a resource is in when it has been activated and is operational. Contrast with *inactive* and *inoperative*. (2) Pertaining to a major or minor node that has been activated by VTAM. Most resources are activated as part of VTAM start processing or as the result of a VARY ACT command.

Active keyboard. A display station enabled to enter data or commands during session mode. The consultant, recipient, moderator, and presenter always have active keyboards.

active link. In RSCS, a teleprocessing link for which a line driver has been initiated; the RSCS operator uses the RSCS START command to initiate a line driver. The link is active until the line driver is terminated by another operator command.

adjacent. In RSCS, two nodes are adjacent if they are connected by only one link.

Advanced Communications Function (ACF). A group of IBM program products (principally VTAM, TCAM, NCP, and SSP) that use the concepts of SNA, including distribution of function and resource sharing.

Advanced Communications Function for Virtual Telecommunications Access Method (ACF/VTAM). An IBM licensed program that controls communications and flow of data in an SNA network. It provides single-domain, multiple-domain, and interconnected network capability.

Advanced Program-to-Program Communications (APPC). The inter-program communication service within SNA LU 6.2 on which the APPC/VM interface is based.

Advanced Program-to-Program Communications/VM (APPC/VM). An API for communicating between two virtual machines that is mappable to the SNA LU 6.2 APPC interface and based on IUCV functions. Along with the TASK virtual machine, AVS virtual machine, and VTAM, APPC/VM provides this communication within a single system, throughout a collection of systems, and throughout an SNA network.

all points addressable (APA). The ability to refer to any pixel on the printable area of the screen. (Pixel is a dot that forms part of an image on the screen.)

alphameric. Synonym for *alphanumeric*.

© Copyright IBM Corp. 1991,1993

alphanumeric. A character set that contains letters, digits, and usually other characters, such as punctuation marks. Synonymous with *alphameric*.

alternate link. In RSCS, a link to transmit data over when the status of none of the primary links defined in the routing of a node is connect. Contrast with *primary link*.

AMS. Access method services.

AP. Attached processor.

APA. All points addressable.

APAR. Authorized program analysis report.

APPC. Advanced Program-to-Program Communications.

APPC link. A logical TSAF link that is physically controlled by VTAM, and is established using the TSAF VTAM line driver.

APPC/VM. Advanced Program-to-Program Communications/VM.

APPC/VM VTAM Support (AVS). A component of VM/ESA that lets application programs using APPC/VM or CPI Communications communicate with programs anywhere in a network defined by IBM's SNA. AVS transforms APPC/VM into APPC/VTAM protocol.

application program. (1) A program written for or by a user that applies to the user's work, such as a program that does inventory control or payroll. (2) A program used to connect and communicate with stations in a network, enabling users to perform application-oriented activities.

apply. When servicing a product or component, to generate an auxiliary control structure from a PTF. See *apply disk*, *apply ID*, *apply message log*, *apply list*, *apply status table*, *apply string*.

apply disk. In VMSES/E, a minidisk or SFS directory containing the files that describe the maintenance levels: the apply status table, AUX files, version vector tables, the select data file, and the build status table.

apply ID. A 1- to 8-character, alphanumeric identifier that is used to name the software inventory files created during apply processing. The user can change this value to define different maintenance levels.

apply list. A file listing PTFs applied to a product or component.

apply message log. A file in which the VMFAPPLY EXEC writes status and error messages during apply processing.

apply status table. The software inventory table that identifies what PTFs have been applied to the product. The system level of the table identifies what product or component has been applied to the system. The file type of the system level inventory table is SYSAPPS and the file type of the service level inventory table is SRVAPPS.

apply string. In VMSES/E, the set of apply disks.

area. A term acceptable for DASD space when there is no need to differentiate between space on count-key-data devices and FB-512 devices. See *DASD space*.

assembler language. A source language that includes symbolic machine language statements in which there is a one-to-one correspondence with instruction formats and data formats of the computer.

attached processor (AP). A processor that has no I/O capability and is always linked to the processor initialized for I/O handling.

Attendee. A conference participant with an inactive keyboard. An attendee views the session and talks with other participants, but cannot enter data or commands.

authority. In SFS, the permission to access a file or directory. You can have read authority or write authority (which includes read authority). You can also have file pool administration authority, which is the highest level of authority in a file pool.

authorized operator. An operator, other than the RSCS central operator, who is authorized to control the RSCS virtual machine or individual links.

authorized program analysis report (APAR). An official request to the responsible IBM Change Team to look into a suspected problem with IBM code or documentation. APARs describe problems giving conditions of failure, error messages, abend codes, or other identifiers. They also contain a problem summary and resolution when applicable. See *program temporary fix (PTF)*.

authorized user ID. In GCS, a user ID that provides access to the GCS supervisor, supervisor state, and (at times) certain restricted CP commands. This access is provided by including the user ID on a list of authorized user IDs compiled with the GCS GROUP EXEC. The virtual machine associated with an authorized user ID is an *authorized* machine, and programs running in that machine are *authorized* applications.

authorized virtual machine. A GCS virtual machine identified by user ID.

auto-signon. An RJE-type link for which you have specified one or more of the auto-signon operands, thus relieving operators of having to manually submit signon cards from remote work stations.

automatic logon. (1) A process when an operator's or user's virtual machine is initiated by someone other than the user of that virtual machine. For example, the primary VM/ESA system operator's virtual machine is automatically started during VM/ESA initialization. A privileged user can enter the AUTOLOG command to start some other (disconnected) virtual machine. (2) A process by which VTAM automatically creates a session-initiation request to establish a session between two LUs. The session will be between a designated primary logical unit (PLU) and a secondary logical unit (SLU) that is neither queued for nor in session with another PLU. See also *controlling application program* and *controlling logical unit*.

automatic software re-IPL. The process by which the control program attempts to restart the system after abnormal termination. This process does not involve the hardware IPL process. See *virtual=real machine recovery*.

AUX file. Auxiliary control file.

auxiliary control file (AUX file). A file that contains a list of file types of update files applied to a particular source file or to control the service level used during build. See *control file* and *preferred auxiliary file*. Synonymous with *auxiliary file*.

auxiliary directory. In CMS, an extension of the CMS file directory for a minidisk, which contains the names and locations of certain CMS modules not included in the minidisk's CMS minidisk file directory.

auxiliary file. Synonym for *auxiliary control file*.

auxiliary storage. Data storage other than main storage; in VM/ESA, auxiliary storage is usually a direct access device.

available. In VTAM, pertaining to an LU that is active, connected, enabled, and not at its session limit.

AVS. APPC/VM VTAM Support.

AVS virtual machine. The virtual machine that manages a gateway that allows APPC communication between VM systems and an SNA network.

B

basic control (BC) mode. A mode in which additional System/370 features, such as new machine instructions, are not operational. Contrast with *extended control (EC) mode*.

binary digit. Either of the digits 0 or 1 when used in the pure binary numeration system. Synonymous with *bit*.

binary synchronous communication (BSC). Communication using binary synchronous line discipline in which transmission of binary-coded data between stations is synchronized by timing signals generated at the sending and receiving stations.

bind. In SNA, a request to activate a session between two LUs. See also *session activation request*. Contrast with *unbind*.

bit. (1) Either of the binary digits 0 or 1. See *byte*. (2) Synonym for *binary digit*.

block. (1) A unit of DASD space on FB-512 devices. For example, FB-512 devices can be the IBM 9335, 9332, 9313, 3370, and 3310 DASD using fixed-block architecture. (2) In CMS Multitasking, to stop the execution of a thread until a function has been completed or a condition is satisfied.

box. In CP, to shut off all communication with an I/O device because CP has identified a serious problem with the device, for example, exceeding the hot I/O rate.

bpi. Bits per inch.

Bpi. Bytes per inch.

BSC. Binary synchronous communication.

buffer. An area of storage, temporarily reserved for performing input or output, from which data is read, or into which data is written. See *forms control buffer (FCB)*, *spool file buffer linkage block*, *terminal input buffer*.

build. (1) In the installation and service of a product, to do the necessary steps to produce executable code or systems. This is often called the *build process*. (2) See *build disk*, *build message log*, *build ID*, *build list*, *build status table*, *build string*.

build disk. In VMSES/E, a minidisk or SFS directory where executable versions of objects serviced are built.

build ID. A 1- to 8-character, alphanumeric identifier that is used to name the software inventory files created during build processing. The user can change this value to define different maintenance levels.

build list. An exec that names the serviceable parts included in an object being built, and parameters and options used to build the object.

build message log. A file in which the VMFBLD EXEC writes status and error messages during build processing.

build status table. The software inventory table that identifies what products have been built, in the system level, and what individual objects have been generated for the product, in the service level. The file type of the system level inventory table is SYSBLDS and the file type of the service level inventory table is SRVBLDS.

build string. The set of build disks.

byte. A unit of storage, consisting of eight adjacent binary digits that are operated on as a unit and constitute the smallest addressable unit in the system.

B1. One of several security designations that can be assigned to a computing system by the U. S. Department of Defense. The B1 designation indicates that the system:

- Restricts access to objects through discretionary access control
- Can audit all events relative to security
- Safely permits the reuse of objects
- Can identify and authenticate each user
- Restricts access to objects through mandatory access control
- Applies a security label to every subject and object.

B1 security criteria include, among other things, all C2 security criteria. See also, *C2*.

C

callable services library (CSL). A package of CMS assembler routines that can be stored as an entity and made available to a high-level language, REXX, or an assembler program.

CAW. Channel address word.

CC. Condition code.

CCS. Console communication service.

CCW. Channel command word.

changes. In installation and service, service supplied by IBM and original equipment manufacturers (OEMs) for their programs. In the IBM service process, there are many ways users can receive information they need to fix (change) a portion(s) of a product they are running on a VM system. These include PTFs, APARs, user modifications, and information received over the phone. All these types of information are called *changes*.

channel. A path in a system that connects a processor and main storage with an I/O device.

channel adapter. A communication controller hardware unit used to attach the controller to a System/360 or a System/370 channel.

channel address word (CAW). An area in storage that specifies the location in main storage at which a channel program begins.

channel command word (CCW). A doubleword at the location in main storage specified by the channel address word. One or more CCWs make up the channel program that directs data channel operations.

channel status word (CSW). An area in storage that provides information about the termination of I/O.

channel-attached. Pertaining to the attachment of devices directly by data channels (I/O channels) to a host processor. Contrast with *link-attached*. Synonymous with *local-attached*.

channel-to-channel (CTC) device. A hardware device that connects two channels on the same computing system or on different systems. CTC devices include both CTCAs and 3088 MCCUs.

channel-to-channel adapter (CTCA). A hardware device that connects two channels on the same computing system or on different systems.

CINIT. A network services request sent from an SSCP to an LU asking that LU to establish a session with another LU and to act as the primary end of the session.

circumventive service. Information that IBM supplies over the telephone or on a tape to circumvent a problem by disabling a failing function until a PTF is available as a corrective service fix. See *patch* and *zap*.

CKD. Count-key-data.

class A user. See *primary system operator privilege class*.

class Any user. Any user, regardless of class, who is allowed to use a subset of CP commands to log onto VM/ESA, have the terminal logically connected to a multiple-access virtual machine, or send messages to the operator or to other users.

class authority. Privilege assigned to a virtual machine user in the user's directory entry; each class specified allows access to a subset of all the CP commands. See *privilege class* and *user class restructure (UCR)*.

class B user. See *system resource operator privilege class*.

class C user. See *system programmer privilege class*.

class D user. See *spooling operator privilege class*.

class E user. See *system analyst privilege class*.

class F user. See *service representative privilege class*.

class G user. See *general user privilege class*.

class override file. A file containing control statements defining changes in the privilege classes of CP commands and diagnose codes. The override program uses this file to establish a new class structure of commands under UCR.

CMS. Conversational Monitor System.

CMS EXEC. An EXEC procedure or EDIT macro written in the CMS EXEC language and processed by the CMS EXEC processor. Synonymous with *CMS program*.

CMS EXEC language. A general-purpose, high-level programming language, particularly suitable for EXEC procedures and EDIT macros. The CMS EXEC processor executes procedures and macros (programs) written in this language. Contrast with *EXEC 2 language* and *Restructured Extended Executor (REXX) language*.

CMS files. Refers exclusively to files in the fixed-block format used by the CMS file system. VSAM and OS data sets and DOS files are not compatible with the CMS file format and cannot be manipulated using CMS file system commands.

CMS minidisk file directory. A directory on each CMS disk that contains the name, format, size, and location of each of the CMS files on that disk. When a disk is accessed by the ACCESS command, its directory is read into virtual storage and identified with any letter from A through Z. Synonymous with *master file directory block* and *minidisk directory*.

CMS nucleus. The portion of CMS that is resident in the user's virtual storage whenever CMS is executing. Each CMS user receives a copy of the CMS nucleus when the user IPLs CMS. See *saved system* and *shared segment*.

CMS program. Synonym for *CMS EXEC*.

CMS system minidisk. The minidisk located at virtual address 190. It contains the system commands for which the user has read-only access.

CMS user disk. One or more virtual disks that contain CMS or read-only DOS or OS files that can be accessed by the user. If the user has read/write access to a disk, the user can create programs and data files on the disk. Files are retained until the user deletes them. The user can also link to and access other users' disks, usually on a read-only basis.

CNTRL file. Control file with file type CNTRL.

command. (1) A request from a user at a terminal for the execution of a particular AVS, CP, CMS, GCS, IPCS, RSCS, or TSAF function. A CMS command can also be the name of a CMS file with a file type of EXEC or MODULE. See *subcommand* and *user-written CMS command*. (2) In SNA, any field set in the transmission header (TH), request header (RH), and sometimes portions of a request unit (RU), that initiates an action or that begins a protocol.

command line. The line at the bottom of display panels that lets a user enter commands or panel selections. It is prefixed by an arrow (= = = = >).

command processor. A program that performs an operation specified by a command.

command response interface (CRI). A utility that lets users specify which method RSCS should use to send command responses, which language RSCS should use to build the message text, and what type of signature RSCS should include with each response. When RSCS sends the command responses, each line of the response is numbered and a message 001 is added to the end of the responses generated for each command.

command string. The represented data that is executed on the host user ID.

commit. (1) In the context of SFS, to change a resource (such as a file) permanently. (2) In the context of CRR, to make permanent changes to protected resources (such as SFS file pools) during a transaction (CRR logical unit of work). CRR commits changes made by an application program or transaction program.

Common Programming Interface (CPI). Provides languages, commands, and calls that allow the development of applications that are more easily integrated and moved across environments supported by SAA. VM/ESA supports the communications and the resource recovery elements of the CPI.

Common Programming Interface (CPI) Communications. A set of program-to-program communication routines that let applications written in REXX and high-level languages access APPC/VM functions. These routines are part of IBM's SAA and are also known as the SAA communications interface.

communication adapter. A program or a part of a program that handles communications with other programs. See *SFS communication adapter*.

communication directory. A CMS facility that lets APPC/VM applications connect to a resource using symbolic destination names and special NAMES files.

communication line. Deprecated term for *telecommunication line* and *transmission line*.

communication link. Synonym for *data link*.

communications directory. A CMS facility that lets APPC/VM applications connect to a resource using symbolic destination names and special NAMES files.

communications server. A virtual machine that provides APPC/VM services between systems within a TSAF collection, and allows for communication between an APPC/VM environment and an SNA-defined network. TSAF and AVS are communication servers. See *intermediate communications server*.

compile. To translate a program written in a high-level programming language into a machine language program.

component. A collection of objects that together form a separate functional unit. A product may contain many components. For example, CP, CMS, and TSAF are components of VM/ESA.

component override. Synonym for *component parameter override*.

component override area. An area of the product parameter file or of a product parameter override file that contains one or more component parameter overrides. Synonymous with *override area*.

component parameter override. A component parameter, defined in a component override area, that updates or replaces a component parameter defined in a component area of the product parameter file. Synonymous with *component override* and *override*.

concurrently. Concerning a mode of operation that includes doing work on two or more activities within a given (short) interval of time.

condition code (CC). A code that reflects the result of a previous I/O, arithmetic, or logical operation.

Conference. A session that can support two to twelve users at separate display stations. The session is controlled by a single moderator; it may have any combination of attendees and presenters.

CONFERENCE CONTROL panel. A panel displayed to the moderator when he presses the PA1 key during a conference session. While this panel is displayed the conference session is suspended. From this panel the moderator can assign a display-input PF key, request the CONFERENCE PARTICIPANTS panel, or pass the PA1 key function to the host or application program for processing.

CONFERENCE PARTICIPANTS panel. A panel the moderator uses during setup to change the status of participants and to start the conference session. The moderator can also access this panel during a conference session to change the status of participants. When the moderator is viewing this panel, the conference session can continue if a presenter also exists. The moderator can return to a session in progress from this panel.

configuration. (1) The arrangement of a computer system or network as defined by the nature, number, and chief characteristics of its functional units. The term may refer to a hardware or a software configuration. (2) The devices and programs that make up a system, subsystem, or network.

configuration file. A file used by RSCS to define the configuration of a computer network.

connect. Establishing a path to communicate with another virtual machine or with the user's own virtual machine.

connected. In VTAM, pertaining to a physical unit (PU) or logical unit (LU) that has an active physical path to the host processor containing the SSCP that controls the PU or LU.

connection. Synonym for *physical connection*.

console. A device used for communications between the operator or maintenance engineer and the computer.

console communication service (CCS). A group of CP modules that interfaces with the VTAM service machine, providing full VM/ESA console capabilities for SNA terminal users.

console spooling. Synonym for *virtual console spooling*.

console stack. Refers collectively to the program stack and the terminal input buffer.

Consult. A session that supports only two users at separate display stations. One user is defined as the consultant and the other user as the recipient.

CONSULT SESSION CONTROL panel. A panel displayed to the consultant when he presses the PA1 key during a consult session. While this panel is displayed the consult session is suspended. From this panel the consultant can assign a display-input PF key or pass the PA1 key function to the host or application program for processing.

Consultant. A user, in a consult session, who provides help or assistance to a recipient. A

consultant always has an active keyboard. He can assign and change a display-input PF key.

contention. The situation where two LUs try to allocate a conversation over the same session at the same time.

continuation character. In RSCS, a character (#) in the configuration file indicating that a statement continues onto the next physical line of text. Using continuation characters, you can split one logical statement onto two or more physical lines. **Note:** When combined, the statement cannot exceed 256 characters.

control block. A storage area that a computer program uses to hold control information.

control file. (1) In service, a file with file type CNTRL that contains records that identify the updates to be applied and the macro libraries, if any, needed to assemble that source program. (2) A CMS file that is interpreted and directs the flow of a certain process through specific steps. For example, the control file could contain installation steps, default addresses, and PTF prerequisite lists and many other necessary items.

control minidisk. In a file pool, the minidisk that tracks the physical DASD blocks allocated to the file pool.

Control mode. The state, in which session mode is suspended, that results when the consultant or moderator presses the PA1 key.

control program. A computer program that schedules and supervises the program execution in a computer system. See *Control Program (CP)*.

Control Program (CP). A component of VM/ESA that manages the resources of a single computer so multiple computing systems appear to exist. Each of these apparent systems, or virtual machines, is the functional equivalent of an IBM System/370, 370-XA, or ESA computer. Also, XC virtual machines provide functions beyond the ESA architecture. See also *virtual machine*.

control program. A computer program that schedules and supervises the program execution in a computer system. See *Control Program (CP)*.

control section (CSECT). The part of a program specified by the programmer to be a relocatable unit, all elements of which are loaded into adjoining main storage.

control statement. A statement that controls or affects program execution in a data processing system.

control terminal. Any terminal currently controlling system resources and used by the primary system operator (privilege class A), the resource operator (privilege class B) or the spooling operator (privilege class D).

control unit. A device that controls I/O operations at one or more devices.

control unit terminal (CUT). An operational mode that allows one logical terminal session. Contrast with *distributed function terminal (DFT)*.

controlling application program. In VTAM, an application program with which a secondary LU (other than an application program) is automatically put in session whenever the secondary LU is available. See also *automatic logon* and *controlling logical unit*.

controlling logical unit. In VTAM, a logical unit with which a secondary LU (other than an application program) is automatically put in session whenever the secondary LU is available. A controlling logical unit can be either an application program or a device-type logical unit. See also *automatic logon* and *controlling application program*.

conversation. A connection between two transaction programs over an LU-LU session that lets them communicate with each other while processing some transaction. The programs establish a conversation, send and receive data in the conversation, and then terminate the conversation.

Conversational Monitor System (CMS). A virtual machine operating system and component of VM/ESA that provides general interactive time sharing, problem solving, program development capabilities, and operates only under the control of the VM Control Program (CP).

Coordinated Resource Recovery (CRR). A CMS facility that implements the LU 6.2 sync point architecture, which ensures that transactions can update multiple protected resources with integrity. This means that all updates, within the transaction, are either completed (committed) or not completed (rolled back or backed out). CRR consists of the coordination function (see *synchronization point processing*), the resynchronization function (see *resynchronization*), and the logging function (see *log minidisks*). The coordination function resides in the application program's virtual machine. The resynchronization and logging functions reside in the CRR recovery server.

copy file. A file having file type COPY that contains nonexecutable real storage definitions that are referred to by macros and assemble files.

copy function. The function initiated by a PF key to copy the contents of a display screen onto an associated hardcopy printer. A remote display

terminal copies the entire contents of the screen onto a printer attached to the same control unit. A local display terminal copies all information from the screen, except the screen status information, onto any printer attached to any local display control unit.

corrective service. Service that IBM supplies on tape to correct a specific problem.

corrective service tape (COR). A tape, supplied by IBM at the user's request, containing a fix for a specific problem and any requisites for the fix.

count-key-data (CKD) device. A DASD that stores data in the format: count field, usually followed by a key field, followed by the actual data of a record. The count field contains the cylinder number, head number, record number, and the length of the data. The key field contains the record's key (search argument).

CP. Control Program.

CP assist. A hardware function, available only on a processor with ECPS, that reduces CP overhead by doing the most frequently used tasks of CP routines.

CP command. A command available to all VM users. Class G CP commands let the general user reconfigure their virtual machine, control devices attached to their virtual machine, do input and output spooling functions, and simulate many other functions of a real computer console. Other CP commands let system operators, system programmers, system analysts, and service representatives manage the resources of the system.

CP directory. Synonym for *VM directory*.

CP read. The condition when CP is waiting for a response or request for work from the user. On a typewriter terminal, the keyboard is unlocked; on a display terminal, the screen status area indicates CP READ.

CP trace table. A table VM/ESA uses for debugging. Its size is a multiple of 4096 bytes and depends on the size of real storage or a user-specified value. This table contains the chronological occurrences of events that take place in the real machine, recorded in a wraparound fashion within the trace table. Synonymous with *trace table*.

CPTRAP. A CP debugging tool that creates a reader spool file of selected trace table entries, CP data, and virtual machine data in the order that they happen. The IPCS commands can help the user access and print this collected data.

CRI. Command response interface.

cross system extensions (CSE). An environment in which end users attached to a single system can participate with additional systems as though all participating systems were one complex.

CRR. Coordinated Resource Recovery.

CRR recovery server. A virtual machine, which is generated during VM/ESA installation, that provides the CRR logging and resynchronization functions. Every processor must have only one CRR recovery server.

CSE. Cross system extensions.

CSL. Callable services library.

CSW. Channel status word.

CTC. Channel-to-channel.

CTCA. Channel-to-channel adapter.

current session. The session that an PVM user has selected.

Customization. The procedure that modifies default values for CVIEW and incorporates changes to the CONSULT IDNAME and DXGKEYS LIST files for the CVIEW product panels.

customization. The process of defining and activating a configuration and changing system processing to meet user requirements.

Customization defaults. The values, stored in the LASTING GLOBALV file and displayed by DXGINIT, that define specific CVIEW functions.

CUT. Control unit terminal.

cylinder. In a disk pack, the set of all tracks with the same nominal distance from the axis about which the disk pack rotates.

C2. One of several security designations that can be assigned to a computing system by the U. S. Department of Defense. The C2 designation indicates that the system:

- Restricts access through discretionary access control
- Can audit all events relative to security
- Safely permits the reuse of objects
- Is able to identify and authenticate each user.

All C2 security criteria are included in B1 criteria. See also *B1*.

D

DASD. Direct access storage device.

DASD space. (1) Area allocated to DASD units on CKD devices. (2) Area allocated to DASD units on FB-512 devices. Note that *DASD space* is synonymous with *cylinder* when there is no need to differentiate between CKD devices and FB-512 devices.

DAT. Dynamic address translation.

data area. A storage area used by a program or device to hold information.

data definition (DD). (1) A program statement that describes the features of, specifies the relationships of, or establishes the context of data. (2) Information that describes the contents and characteristics of a field, record, or file. A data definition can include such things as field names, lengths, locations, and data types.

data definition (DD) name. The name of a date definition (DD) statement that corresponds to a data control block that contains the same name.

data definition (DD) statement. A job control statement describing a data set associated with a specific job step.

data link. (1) The equipment and rules (protocols) used for sending and receiving data. Synonymous with *communication link*. (2) In SNA, synonym for *link*.

data stream. A set of logical records sent one after the other.

DBCS. Double-byte character set.

DCSS. Discontiguous saved segment.

DD. Data definition.

DDP. Distributed data processing.

deactivate. To take a resource of a node out of service, rendering it inoperable, or to place it in a state in which it cannot perform the functions for which it was designed. Contrast with *activate*.

dedicated device. An I/O device or line not being shared among users. The facility can be permanently assigned to a particular virtual machine by a VM/ESA directory entry, or temporarily attached by the resource operator to the user's virtual machine.

dedicated maintenance mode. In a file pool server machine, a mode of file pool server processing during which the file pool server machine has exclusive use of the file pool. The file pool is unavailable to other users. Contrast with *multiple user mode*.

default. An attribute, value, or option that is assumed when none is explicitly specified.

default operand. An operand that has a preset value if a value is not specified on the CP or CMS command line.

delimiter. (1) A flag that separates and organizes items of data. Synonymous with *separator*. (2) A character used to indicate the beginning and end of a character string. (T) (3) A character that groups or separates words or values in a line of input.

device. (1) A mechanical, electrical, or electronic contrivance with a specific purpose. (2) See *channel-to-channel (CTC) device*, *count-key-data (CKD) device*, *dedicated device*, *device block*, *device driver*, *device support facilities*, *direct access storage device (DASD)*, *display device*, *draining a unit record device*, *fixed-block architecture (FBA) device*, *primary paging device*, *spooling devices*, *unit record device*, *virtual spooling device*.

device block. A storage area used by a computer program to hold device information.

device support facilities. A program for doing operations on disk volumes so that they can be accessed by IBM and user programs. Examples of these operations are initializing a disk volume and assigning an alternate track.

DFT. Distributed function terminal.

DIAL command. The CP command that accesses the CVIEW virtual machine to begin setup for a consult or conference session.

DIRCONTROL directory. Synonym for *directory control directory*.

direct access storage device (DASD). A storage device in which the access time is effectively independent of the location of the data.

directory. See *auxiliary directory*, *CMS minidisk file directory*, *directory control directory*, *file control directory*, *SFS directory*, or *VM directory*.

directory control directory. A type of SFS directory with functional characteristics like a minidisk's. A single access authority applies to the directory and all the files in the directory. When you access a directory control directory in read-only mode, you cannot see changes made until you release and reaccess the directory. When you access the directory in read/write mode, changes become available as they are made. Synonymous with *DIRCONTROL directory*. Contrast with *file control directory*.

disabled. (1) The state of an RSCS exit that prevents the exit from receiving control from RSCS. (2) In VTAM, pertaining to an LU that has indicated to its SSCP that it is temporarily not ready to establish LU-LU sessions. An initiate request for a session with a disabled LU can specify that the session be queued by the SSCP until the LU becomes enabled. The LU can separately indicate whether this applies to its ability to act as a primary LU (PLU) or a secondary LU (SLU). See also *enabled* and *inhibited*.

disconnect. (1) To disengage apparatus used in a connection and restore it to its ready condition when not in use. Synonymous with *release*. (2) To break a connection, physically or electrically.

disconnect mode. The mode of operation in which a virtual machine is executing without a physical line or terminal connected as an operator console. Any attempt to issue a read to the console causes the virtual machine to be logged off after 15 minutes have elapsed, unless the user logs on again within the 15-minute interval. Note that with the SCIF, a user can be disconnected from a primary virtual console but still have console communications through the console of the secondary user.

discontiguous saved segment (DCSS). A saved segment that begins and ends on a megabyte boundary and is not a segment space or a member of a segment space. A DCSS may contain logical saved segments. Contrast with *segment space* and *member saved segment*.

disk. A magnetic disk unit in the user's CMS virtual machine configuration. See *virtual disk*.

disk operating system (DOS). An operating system for computer systems that use disks and diskettes for auxiliary storage of programs and data.

dispatcher. The program in CP that places virtual machines or CP tasks into execution. The dispatcher selects the next virtual machine to run and prepares the virtual machine for problem state execution.

dispatching. (1) In VM/ESA, the starting of virtual machine execution. See *dispatching class*.

dispatching class. A set of threads with two properties:

- No thread in the class is ever preempted by another in the same class unless the first thread voluntarily ceases execution.
- No two threads in a class are ever processed in parallel.

However, any thread can be preempted by or processed in parallel with any thread in a different class.

display. (1) To present information for viewing, usually on a terminal screen or a hardcopy device. (2) A device or medium on which information is presented, such as a terminal screen. (3) Deprecated term for *panel*.

display device. An I/O device that gives a visual representation of data.

display mode. A type of editing at a display terminal in which an entire screen of data is displayed at once and in which the user can access data through commands or by using a cursor. Contrast with *line mode*.

Display station. A 3270 device type that consists of a keyboard and display.

display terminal. A terminal with a component that can display information on a viewing surface such as a screen or gas panel.

Display-input option. A function that sends data, commands, and the cursor position to all display stations in a session before the host or application program processes them.

Display-input PF key. The program function key that invokes the display-input function when pressed from an active keyboard during a CVIEW session.

distributed data processing (DDP). Data processing in which processing, storage, and control functions, in addition to I/O operations, are distributed among remote locations and connected by transmission facilities.

distributed function terminal (DFT). An operational mode that allows multiple concurrent logical terminal sessions. Contrast with *control unit terminal (CUT)*.

distributed processing. A transaction that involves more than one node to complete its processing. A node might be a VM/ESA virtual machine in the same processor or a different processor or it may be an LU partner in a non-VM system (for example, MVS). We can think of the use of SFS as distributed processing (a user machine and a server machine cooperating) and the file pool as a distributed resource.

DOS. Disk operating system.

double-byte character set (DBCS). A character set that requires 2 bytes to uniquely define each character. This contrasts with EBCDIC, in which each printed character is represented by 1 byte.

draining a unit record device. The stopping of spooling operations on a real unit record device after the current spool file has completed, or the immediate stopping of spooling operations (if no spool file is being processed).

dump. To write the contents of part or all of main storage, or part or all of a minidisk, to auxiliary storage or a printer. See *abend dump*.

Dump Viewing Facility. A component of VM/ESA that lets users display, format, and print data interactively from CP hard and soft abend, stand-alone, and virtual machine dumps; process CP trace table data stored on tape or in a system trace file; and display symptom records.

dynamic address translation (DAT). In System/370 and ESA virtual storage systems, the change of a virtual storage address to a real storage address during execution of an instruction.

dynamic paging area (DPA). An area of real storage that CP uses for virtual machine pages and pageable CP modules.

E

EBCDIC. Extended binary-coded decimal interchange code.

ECPS:VM/370. Extended Control Program Support:VM/370.

edit. A function that makes changes, additions, or deletions to a file on a disk. These changes are interactively made. The edit function also generates information in a file that did not previously exist.

element. In reference to installation and service of a product, a file provided on a program update tape (PUT) as input to the build process. An element is the smallest serviceable unit of a component. Several files can be associated with a given element, but each file has the same file name. See *build*.

emulation. The use of programming techniques and special machine features to permit a computing system to execute programs written for another system.

emulation program (EP). A control program that lets an IBM 3704 or 3705 Communications Controller emulate the functions of an IBM 2701 Data Adapter Unit, an IBM 2702 Transmission Control Unit, or an IBM 2703 Transmission Control Unit.

enable. Deprecated term for removing a lock from an object in the CMS Shared File System, in particular from a file space or storage group.

enabled. (1) The state of an RSCS exit that allows the exit to receive control from RSCS. (2) In VTAM, pertaining to an LU that has indicated to its SSCP that it is now ready to establish LU-LU sessions. The LU can separately indicate whether this prevents it from acting as a primary LU (PLU) or as a secondary LU (SLU). See also *disabled* and *inhibited*.

end user. In SNA, the ultimate source or destination of application data flowing through an SNA network. An end use may be an application program or a terminal operator.

end-of-transmission (EOT). The specific character, or sequence of characters, that indicates no more data.

entry point. An address or label of an instruction performed on entering a computer program, a routine, or a subroutine. A program can have several different entry points, each corresponding to a different function or purpose.

environmental record editing and printing program (EREP). A program that makes the data contained in the system recorder file available for further analysis.

EOF. End of file.

EOT. End-of-transmission.

EP. Emulation program.

EREP. Environmental record editing and printing program.

ESA virtual machine. A virtual machine that simulates ESA/370 or ESA/390 functions. Contrast with *370 virtual machine*, *XA virtual machine*, and *XC virtual machine*

escaped command. A command that is executed on the user's host user ID, rather than on a PVM session user ID.

ESCON. Enterprise systems connection.

event. (1) An occurrence of significance to a task; for example, the completion of an asynchronous operation, such as an input/output operation.

exclusive segment. A range of pages within a saved system, member saved segment, or discontiguous saved segment that can be accessed by only one virtual machine at a time.

exit. See *user exit* and *installation-wide exit*.

exit point. An instruction in a program, routine, or subroutine that causes control to pass to another program, routine, or subroutine. After the other program, routine, or subroutine has finished processing, control usually returns to the instruction in the original program, routine, or subroutine that caused control to pass.

exit routine. (1) A routine that receives control when a specified event occurs, such as an error. (2) Any of several types of user-written routines. See *ASCII printer plotter exits*, *RSCS exit facility*, and *transmission algorithms*. (3) In PVM, a macro or exec that PVM users can write to customize their PVM environment. PVM modules at certain points

temporarily stop their own processing and branch to exit routines.

expanded virtual machine assist. A hardware assist function, available only on a processor that has ECPS, that handles many privileged instructions not handled by VMA, and extends the level of support of certain privileged instructions beyond that provided by VMA.

explicit route (ER). In SNA, the path control network elements, including a specific set of one or more transmission groups, that connect two subarea nodes. An explicit route is identified by an origin subarea address, a destination subarea address, an explicit route number, and a reverse explicit route number. Contrast with *virtual route*. See also *path* and *route extension*.

extended binary-coded decimal interchange code (EBCDIC). A set of 256 characters, with each character represented by 8 bits.

extended color. A capability that allows color terminals to display fields or characters in colors using extended field and character attributes.

extended control (EC) mode. A mode in which all features of a System/370 computing system, including dynamic address translation, are operational. Contrast with *basic control (BC) mode*.

Extended Control Program Support (ECPS:VM/370). A hardware assist feature that improves the performance of CP by reducing CP overhead. ECPS:VM/370 consists of CP assist, expanded virtual machine assist, and virtual interval timer assist.

extended data stream (EDS). A 3270 data stream that contains extended highlighting, extended color, or programmed symbols (or a combination of those).

extended highlighting. A function that provides blinking, reverse video, and underscore for emphasizing display fields or characters. This only pertains to devices that support extended filed attributes and character attributes.

extended PLIST (untokenized parameter list). Four addresses that indicate the extended form of a command as it was entered at a terminal.

external page storage. The portion of auxiliary storage that contains 4K-byte pages.

external page storage management. A set of routines in the paging supervisor that control the transfer of data between real storage and external page storage.

F

facility. A service provided by an operating system for a particular purpose.

FBA. Fixed-block architecture.

FCB. (1) Forms control buffer. (2) Function control block. (3) File control block.

feature. A part of an IBM product that can be ordered separately by the customer.

FIFO (first-in-first-out). A queuing technique in which the next item to be retrieved is the item that has been on the queue for the longest time. Contrast with *LIFO (last-in-first-out)*.

file access mode. A file mode number that designates whether the file can be used as a read-only or read/write file by a user. See also *file mode*.

file control directory. A type of SFS directory for which separate access authorities are granted to the directory and to the individual files in the directory. When you access a file control directory, changes to the directory become available as they are made. Synonymous with *FILECONTROL directory*. Contrast with *directory control directory*.

file ID. A CMS file identifier that consists of a file name, file type, file mode, or directory ID. The file ID is associated with a particular file when the file is created, defined, or renamed under CMS. See *file name, file type,* and *file mode*.

file mode. (1) A two-character CMS file identifier containing the file mode letter (A through Z) followed by the file mode number (0 through 6). The file mode letter indicates the minidisk or SFS directory on which the file resides. The file mode number indicates the access mode of the file. (2) A one-character CP file identifier containing the file mode letter (A through Z) of a disk that CP has accessed. (3) See also *file access mode*.

file name. A one-to-eight character alphanumeric field, containing A through Z, 0 through 9, and special characters $ # @ + - (hyphen) : (colon) _ (underscore), that is part of the CMS file identifier and serves to identify the file for the user.

file type. A one-to-eight character alphanumeric field, containing A through Z, 0 through 9, and special characters $ # @ + - (hyphen) : (colon) _ (underscore), that is used as a descriptor or as a qualifier of the file name field in the CMS file identifier. See *reserved file types*.

FILECONTROL directory. Synonym for *file control directory*.

fixed-block architecture (FBA) device. A disk storage device that stores data in blocks of fixed size or records; these blocks are addressed by block number relative to the beginning of the particular file.

flush list. A set of pages available to replenish the free list.

forms control buffer (FCB). In the 3800 Printing Subsystem, a buffer for controlling the vertical format of printed output. The FCB is analogous to the punched-paper, carriage-control tape that IBM 1403 Printers use.

free storage. Storage not allocated. The blocks of central storage available for temporary use by programs or by the system.

full recording mode. The mode of operation where transient processor and main storage errors that are corrected or circumvented by hardware retry or error correction code logic, are then recorded on the VM/ESA error recording cylinders.

full-pack minidisk. A virtual disk that contains all of the addressable cylinders of a real DASD volume.

function control block (FCB). In Subsystem Support Services (SSS), a control block that contains information such as a function's status, event control block, task I/O queue, and I/O queue.

G

gateway. (1) The combination of machines and programs that provide address translation, name translation, and SSCP rerouting between independent SNA networks to allow those networks to communicate. A gateway consists of one gateway NCP and at least one gateway SSCP. (2) In VM, the LU name of a VM system or TSAF collection that is a source for communications to an SNA-defined network or the target of communications from an SNA-defined network.

gateway NCP. An NCP that performs address translation to allow cross-network session traffic. The gateway NCP connects two or more independent SNA networks.

gateway node. Synonym for *gateway NCP*.

GCS. Group Control System for ESA/370 or ESA/390 architecture.

GCS370. Group Control System for CP/370 architecture.

General commands. The CVIEW commands that can be issued by any user from their own virtual machine console to the CVIEW virtual machine.

general register. In CMS, a register that does operations such as binary addition, subtraction, multiplication, and division. General registers primarily compute and modify addresses in a program.

general user privilege class. The subset of CP commands that lets the Class G user manipulate and control a virtual machine.

generation. The process of assembling and link editing definition statements so that resources can be identified to all the necessary programs in a network.

generic bind. Synonym for *session activation request*

generic unbind. Synonym for *session deactivation request*.

global gateway. (1) A gateway that programs outside a CS or TSAF collection can use to access global resources inside the collection. (2) A gateway that global resource manager programs use to access resources outside a CS or TSAF collection. Contrast with *private gateway* and *system gateway*.

global resource. A resource accessible from anywhere within a CS or TSAF collection and whose identity is known throughout the collection. A shared file system file pool is an example of a global resource. Contrast with *local resource*, *private resource*, and *system resource*.

global variable. A PVM variable whose value applies to all of a user's active sessions.

GRAF address. A virtual address assigned to a GRAF by the CVIEW virtual machine. The valid GRAF addresses for the CVIEW virtual machine are 501 through 530.

GRAF device. A 3270 display station that has accessed the CVIEW virtual machine by issuing the DIAL CVIEW command.

group. (1) In RSCS, synonym for *routing group*. (2) In VM/ESA, synonym for *virtual machine group*.

group configuration file. A file that the GROUP EXEC creates. It contains the *blueprint* for building the user's virtual machine group. The name of the file is *systemname* GROUP, where *systemname* is the name of the user's GCS saved system.

Group Control System (GCS). A component of VM/ESA, consisting of a shared segment that the user can IPL and run in a virtual machine. It provides simulated MVS services and unique supervisor services to help support a native SNA network.

Group Control System (GCS370). The component of the 370 Feature of VM that, as a virtual machine supervisor, executes in a group of System/370 virtual

machines under CP370 control. This provides an interface that helps support a native SNA network.

guest. An operating system running in a virtual machine managed by a VM control program. Contrast with *host*.

guest virtual machine (GVM). A virtual machine in which an operating machine is running.

GVM. Guest virtual machine.

H

hardcopy. A printed copy of machine output in a visually readable form; for example, printed reports, listings, documents, summaries, or network logs.

help function. One or more display images that describe how to use application software or how to do a system operation.

help panel. An online display that tells you how to use a command or another aspect of a product. See *task panel*.

High Performance Option (HPO). A program product that is an extension of VM/SP. It provides performance and operation enhancements for large system environments.

HOME panel. The first panel displayed during CVIEW setup mode. Users select a session role (recipient, consultant, moderator, or attendee) from the HOME panel.

Home PF key. The program function key that causes the display of the HOME panel when pressed in response to another menu or data-entry panel.

host. A VM control program in its capacity as manager of a virtual machine in which another operating system is running. Contrast with *guest*.

host session. The CMS session from which a user initially invokes PVM.

host system. A data processing system that prepares programs and the operating environments for use by another computer or controller.

HPO. High Performance Option.

I

I/O. Input/output.

I/O Configuration Program (IOCP). A program that defines to a system all available I/O devices and channel paths.

IBM-defined exit. An exit point that has already been placed and designed into the RSCS code by IBM.

ID card. Under VM/ESA, the identification card that indicates the destination user ID of a deck of real cards. These cards are read into the system card reader or into the card reader of an RSCS remote station.

imbed. (1) To make something an integral part of. (2) To place or fix firmly in surrounding matter.

imbedded file. A file that is included into another file. Contrast with *master file*.

IML. Initial microprogram load.

in-doubt. A protected resource is called in-doubt when it has successfully completed the first phase of the two-phase commit and it is waiting for a decision from the initiator to either commit or roll back the changes and therefore start the second phase of the two-phase commit.

in-queue virtual machine. A virtual machine on the run list waiting to be dispatched.

inactive. In VTAM, describes the state of a resource that has not been activated or for which the VARY INACT command has been issued. Contrast with *active*. See also *inoperative*.

Inactive keyboard. A display station that is unable to enter data or commands during a CVIEW session. An attendee in a conference session always has an inactive keyboard.

inactive page. A page in real storage that has not been referenced during a predetermined period.

inhibited. In VTAM, pertaining to an LU that has indicated to its SSCP that it is not ready to establish LU-LU sessions. An initiate request for a session with an inhibited LU will be rejected by the SSCP. The LU can separately indicate whether this applies to its ability to act as a primary LU (PLU) or as a secondary LU (SLU). See also *enabled* and *disabled*.

initial microprogram load (IML). The action of loading a microprogram, which is a set of elementary instructions maintained in special storage that corresponds to a specific computer operation.

initial program load (IPL). The initialization procedure that causes an operating system to begin operation. A VM user must IPL the specific operating system into the virtual machine that will control the user's work. Each virtual machine can be loaded with a different operating system. See *IPL parameters*, *IPL processor*.

initialize. To set counters, switches, addresses, or contents of storage to starting values.

initiate. A network services request sent from an LU to an SSCP requesting that an LU-LU session be established.

inoperative. The condition of a resource that has been active, but is not. The resource may have failed, received an INOP request, or is suspended while a reactivate command is being processed. See also *inactive*.

input line. For typewriter terminals, information entered by a user between the time the typing element of the terminal comes to rest following a carriage return until another carriage return is typed. For display terminals, the data entered into the user input area of the screen. See *user input area*.

input/output (I/O). (1) A device whose parts can do an input process and an output process at the same time. (2) A functional unit or channel involved in an input process, output process, or both, concurrently or not, and to the data involved in such a process.

installation-wide exit. An interface to VM/ESA that a system programmer can use to enhance or extend the functions of a VM/ESA system. Generally, an installation-wide exit is activated for all users on the system and is run as part of a system program.

Installed options. The values stored in the LASTING GLOBALV, CONSULT IDNAME, and DXGKEYS LIST files by the CVIEW installation exec.

installed user program (IUP). An IBM licensed program produced by or for an individual IBM system user (customer or IBM internal). This kind of program can interface with licensed programs, or it can be a stand-alone program.

intensive recording mode. A special error recording mode that can be invoked by an IBM service representative for only one I/O device at a time. On the 1st through 10th unit checks or other error conditions specified by the service representative, an I/O error record is constructed, formatted, and written to the VM/ESA I/O error recording cylinder, after which no more errors are recorded.

inter-user communication vehicle (IUCV). A VM/ESA generalized CP interface that helps the transfer of messages either among virtual machines or between CP and a virtual machine.

interaction. A basic unit that records system activity, consisting of acceptance of a line of terminal input, processing of the line, and a response, if any.

interactive. The classification given to a virtual machine depending on this virtual machine's processing characteristics. When a virtual machine uses less than its allocation time slice because of terminal I/O, the virtual machine is classified as being interactive. Contrast with *noninteractive*.

Interactive Problem Control System (IPCS). A component of VM that permits online problem management, interactive problem diagnosis, online debugging for disk related CP or virtual machine abend dumps or CPTRAP files, problem tracking, and problem reporting.

Interactive Problem Control System (IPCS). A component of VM/ESA (370 Feature) that permits online problem management, interactive problem diagnosis, online debugging for disk related CP or virtual machine abend dumps or CPTRAP files, problem tracking, and problem reporting.

Interactive session. The activities that occur from the time a user logs on a system until the time a user logs off the system.

interactive user. A user whose virtual machine is dispatched for the first time, or whose virtual machine had an I/O interrupt from the terminal before the previous queue had ended.

interface. A shared boundary between two or more entities. An interface might be a hardware or software component that links two devices or programs together.

intermediate communications server. A virtual machine that handles communications requests to a resource manager program for a user program.

internal trace table. See *CP trace table*.

interrupt. A suspension of a process, such as execution of a computer program, caused by an external event and done in such a way that the process can be resumed.

invoke. To start a command, procedure, or program.

IOCP. I/O Configuration Program

IPCS. Interactive Problem Control System.

IPL. Initial program load

IPL processor. In an AP or MP system, the processor on which the control program was first initialized during system generation. Note that both the IPL and the non-IPL processors in a real MP configuration have I/O capabilities.

IUCV. Inter-user communication vehicle

IUP. Installed user program.

K

K. Kilobyte.

keyword. (1) A name or symbol that identifies a parameter. (2) A part of a command operand that consists of a specific character string (such as FEATURE=).

L

label. (1) In programming languages, a language construction naming a statement and including an identifier. (I) (2) An identifier of a command generally used for branching. (3) See *label definition*, *label reference*.

LAN. Local area network.

LIFO (last-in-first-out). A queuing technique in which the next item to be retrieved is the item most recently placed in the queue. Contrast with *FIFO (first-in-first-out)*.

line. See *communication line*.

line driver. An RSCS task (program) that permits communication between RSCS and a non-SNA type of remote station. Contrast with *session driver*.

line group. One or more telecommunication lines of the same type that can be activated and deactivated as a unit.

line mode. The mode of operation of a display terminal that is equivalent to using a typewriter-like terminal. Contrast with *display mode*.

line number. A number located at either the beginning or the end of a record (line) that can be used during editing to refer to that line. See *prompting*.

line speed. The number of binary digits that can be sent over a telecommunication line in one second, expressed in bits per second (bps).

link. (1) In RSCS, a connection, or ability to communicate, between two adjacent nodes in a network. (2) In TSAF, the physical connection between two systems. (3) In SNA, the combination of the link connection and the link stations joining network nodes; for example: (1) a System/370 channel and its associated protocols, (2) a serial-by-bit connection under the control of Synchronous Data Link Control (SDLC). A link connection is the physical medium of transmission. A link, however, is both logical and physical. Synonymous with *data link*.

link ID. Link identifier.

link table. An RSCS control block that contains the definitions for all the links in a system.

link-attached. In VTAM, pertaining to devices that are physically connected by a telecommunication line. Synonymous with *remote*. Contrast with *channel-attached*.

load. In installation and service, to move files from tape to disk, auxiliary storage to main storage, or minidisks to virtual storage within a virtual machine.

load map. A map containing the storage addresses of control sections and entry points of a program loaded into storage.

load module. A computer program in a form suitable for loading into main storage for execution.

local. (1) Two entities (for example, a user and a server) are said to be local to each other if they belong to the same system within a collection or to the same node within an SNA system. (2) In RSCS, channel-attached; that is, connected through only a channel, not through an RSCS-defined link. Local refers to a computer, I/O devices, and virtual machines that communicate with one another only over channels. Contrast with *remote*.

local address. In SNA, an address used in a peripheral node in place of an SNA network address and transformed to or from an SNA network address by the boundary function in a subarea node.

local area network (LAN). A data network located on the user's premises in which serial transmission is used for direct data communication among data stations. Contrast with *wide area network (WAN)*.

Local display station. The 3270 device type architecture that is supported by CVIEW Version 2 and the CP DIAL command.

local ID. (1) In RSCS, the node ID of the user's local system. (2) In SFS, the user ID on your local system of some remote user.

local modification. Synonym for *local service*.

local node ID. Synonym for *local ID*.

local resource. (1) A resource accessible from only within a single VM/ESA system and whose identity is known only within a single VM/ESA system in the CS or TSAF collection. (2) A resource accessible from only within a single OS/2 Extended Edition or AIX PS/2 workstation running VM PWSCS. Contrast with *global resource*, *private resource*, and *system resource*.

local service. (1) Change applied to a product without using VMSES/E. (2) See also *circumventive service* and *user modification*. (3) Synonymous with *local modification*.

local-attached. Depreciated term for *channel-attached*.

locked page. A page that is not to be paged out.

log minidisks. (1) In SFS, two duplicate minidisks that contain information about changes made to the file pool. SFS file pool servers use the SFS log minidisks to help protect the integrity of the file pool if a system failure occurs. (2) In CRR, two duplicate minidisks that contain information about the states of various logical units of work during sync point processing. CRR recovery servers use the CRR log minidisks to perform resynchronization processing to protect the integrity of the transaction if a failure occurs during sync point processing.

log off. To request that a session be terminated.

log on. (1) To initiate a session. (2) In SNA, to initiate a session between an application program and a logical unit (LU).

logic error. In VTAM, an error condition that results from an invalid request; a program logic error.

Logical device. A local display station that has no real existence to CP and is driven by the CVIEW virtual machine. Each GRAF device in a consult or conference session is associated with a single logical device.

Logical device address. A virtual address assigned by CP to a logical device created for a consult or conference session. The address is expressed as Lnnn, where L is constant and nnn represents three hexadecimal digits.

logical record. A formatted record that consists of a 2-byte logical record length and a data field of variable length.

logical saved segment. A saved segment defined by CMS within a physical saved segment. A single physical saved segment can contain many logical saved segments. A logical saved segment can contain different types of program objects, such as MODULE files, TEXT files, execs, callable services libraries, language information files, user-defined objects, or a single minidisk directory. See also *physical saved segment*.

logical segment definition file. A file that identifies the contents of a logical saved segment.

logical unit (LU). (1) In SNA, a port through which an end user accesses the SNA network to communicate with another end user and through which the end user accesses the functions provided by system services control points (SSCPs). An LU can support at least two sessions, one with an SSCP and one with another LU, and may be capable of supporting many sessions with other logical units. (2) In VM, an entity addressable within an SNA-defined network, similar to a node within a VM network. LUs are categorized by the types of communication they support. A TSAF collection in an SNA network is viewed as one or more LUs.

logical unit name (LU name). A symbolic name given to a particular LU in an SNA-defined network.

logical unit of work. (1) In SFS, a group of related operations that the SFS file pool server is doing for a user. The operations in a logical unit of work can either be committed or rolled back as a unit. Sometimes this is called a resource logical unit of work. (2) In CRR, a logical unit of work is a convenient abstraction for the application processing (including the underlying system support) performed to take a set of protected resources (such as SFS file pools) from one consistent state to another (commit changes) in such a way that the unit of work appears atomic. If a failure occurs during the sync point processing of a logical unit of work, any changes made by (or for) the logical unit of work are rolled back, so that the protected resources are returned to their previous consistent state. A CRR logical unit of work is frequently called a *transaction* or *LUWID* and consists of one or more *LUWID instances*.

logical unit of work identifier (LUWID). The identifier of a CRR logical unit of work. The LUWID includes three parts: the fully qualified LU network name; the instance number, which is unique at the LU that creates it; and the sequence number, which is incremented by one following a sync point. Also, the conversation correlator is used to further qualify LUWIDs.

logically connected terminal. A terminal connected by leased or switched telecommunications or by local attachment to a multiple-access virtual machine by the CP DIAL command.

logoff. (1) The procedure by which a user ends a terminal session. (2) In VTAM, an unformatted session termination request.

logon. (1) The procedure by which a user begins a terminal session. (2) In VTAM, an unformatted session initiation request for a session between two logical units. See *automatic logon* and *simulated logon*. See also *session-initiation request*.

logon mode. In VTAM, a subset of session parameters specified in a logon mode table for communication with a logical unit. See also *session parameters*.

logon mode table. In VTAM, a set of entries for one or more logon modes. Each logon mode is identified by a logon mode name.

LU. Logical unit.

LU name. Logical unit name.

LU type. Deprecated term for *LU-LU session type*.

LU-LU session. In SNA, a session between two logical units (LUs) in an SNA network. It provides communication between two end users, or between an end user and an LU services component.

LU-LU session type. In SNA, the classification of an LU-LU session in terms of the specific subset of SNA protocols and options supported by the logical units (LUs) for that session, namely:

- The mandatory and optional values allowed in the session activation request.
- The usage of data stream controls, FM headers, request unit (RU) parameters, and sense codes.
- Presentation services protocols such as those associated with FM header usage.

LU-LU-session types 0, 1, 2, 3, 4, 6, and 7 are defined.

LUWID. Logical unit of work identifier.

M

MAC. mandatory access control.

machine. A synonym for a virtual machine running under the control of VM/ESA.

machine ID. A 2-byte field that uniquely defines a virtual machine within a virtual machine group. Machine ID is sometimes combined with task ID to uniquely identify a task within the virtual machine group.

macro. Abbreviation for *macrodefinition* and *macroinstruction*.

macro library. A library of macrodefinitions.

macrodefinition. A set of statements that defines the name of, format of, and conditions for generating a sequence of assembler language statements from a single source statement. Synonymous with *macro*.

macroinstruction. In assembler language programming, an assembler language statement that causes the assembler to process a predefined set of statements called a macrodefinition. The statements usually produced from the macrodefinition replace the macroinstruction in the program. Synonymous with *macro*.

major node. In VTAM, a set of resources that can be activated and deactivated as a group. See *node* and *minor node*.

mandatory access control (MAC). A security regimen that governs which subjects can access which objects, and in what way, based upon the relationship between their security labels. Mandatory access control restricts a subject's access to an object based upon three things:

1. The security label of the subject
2. The security label of the object
3. The type of access the subject wants.

map. (1) In CMS, the file that contains a CMS output listing, such as (i) a list of macros in the MACLIB library, including macro size and location within the library; (ii) a listing of the directory entries for the DOS/VS system or private source, relocatable, or core image libraries; (iii) a linkage editor map for CMS/DOS programs; and (iv) a module map containing entry point locations. (2) A set of values having defined correspondence with the quantities or values of another (3) To show relationships between objects.

master file. A file containing statements that imbed other files into it. Contrast with *imbedded file*.

master file directory. A directory on each CMS disk that contains the name, format, size, and location of all the CMS files on the disk. When a disk is accessed by the ACCESS command, the directory is read into main storage and identified with one of the 26 disk mode letters (A through Z).

master file directory block. Synonym for *CMS minidisk file directory*.

MDISK. (1) Another name for minidisk. (2) The VM directory statement that describes a user's storage space.

megabyte (MB). 1,048,576 bytes.

member saved segment. A saved segment that begins and ends on a page boundary and belongs to up to 64 segment spaces. A member saved segment is accessed by its own name or by the name of a segment space to which it belongs. A member saved segment may contain logical saved segments. Contrast with *discontiguous saved segment*.

memo-to-users. (1) In VM, a file provided on a service tape that contains specific service information for a product. (2) In RSCS, a file provided on the base tape, feature tape, or service tape that contains specific information about installing or servicing RSCS.

message. (1) Data sent from a source application to a target application program in a conversation. (2) In VTAM, the amount of FM data transferred to VTAM by the application program with one SEND request.

message header. The leading part of a message that contains information such as the source or destination code of the message, the message priority, and the type of message.

message repository. A source file that contains message texts for a VM component or user application. It is compiled into internal form by the GENMSG command. The message text in a repository file can be translated and used to support national languages.

message text. The part of a message of concern to the party ultimately receiving the message, that is, the message exclusive of the header or control information.

migration. Installing a new version or release of a program when an earlier version or release is already in place.

minidisk. (1) A logical subdivision (or all) of a physical disk pack that has its own virtual device address, consecutive virtual cylinders (starting with virtual cylinder 0), and a VTOC or disk label identifier. Each user virtual disk is preallocated and defined by a VM/ESA directory entry as belonging to a user. (2) See *CMS minidisk file directory*, *CMS system minidisk*, *control minidisk*, *full-pack minidisk*,*log minidisks*, *minidisk directory*, *minidisk pool*, *minidisk-relative block number*.

minidisk directory. Synonym for *CMS minidisk file directory*.

minidisk pool. A logical DASD containing one or more DASD or minidisk extents that are formatted with 4K records. It is mapped to address spaces.

minidisk-relative block number. A 4K record contained on a minidisk or DASD extent with its origin relative to zero.

minor node. In VTAM, a uniquely-defined resource within a major node. See *node* and *major node*.

mode name. Part of the CPI Communications side information. The mode name is used by LU 6.2 to designate the properties for the session that will be allocated for a conversation.

Moderator. The CVIEW user who controls a conference session. The moderator starts the session, changes participants' status, drops participants, and assigns the display-input PF key. The moderator always has an active keyboard.

module. (1) A program unit that is discrete and identifiable with respect to compiling, combining with other units, and loading; for example, the input to or output from an assembler, compiler, linkage editor, or executive routine. (A) (2) A nonrelocatable file whose external references have been resolved.

MPVM. VM/Pass-Through Facility Multiple Session Application

multiple preferred guests. A facility that supports up to six preferred virtual machines when the Processor Resource/Systems Manager (PR/SM) feature is installed in the real machine. See *preferred virtual machine*.

multiple user mode. In a file pool server machine, a mode of file pool server processing during which it processes user requests for file pool data. (The CMS FILESERV START command starts multiple user mode processing.) Contrast with *dedicated maintenance mode*.

Multiple Virtual Storage (MVS). An alternative name for OS/VS2.

multiple-access virtual machine. A virtual machine running under VM/ESA that supports teleprocessing terminals.

multisystem channel communications unit (MCCU). An I/O device; for example, the 3088 MCCU, which interconnects up to eight systems using block-multiplexer channels.

multitasking. Providing services for many tasks that are active at the same time.

MVS. Multiple Virtual Storage.

MVS page fault assist. A standard hardware feature on the IBM 3081 processor complex that reduces the time needed by the MVS/SP virtual machine to handle page translation exception interruptions. The MVS/SP V=R guest gets these interruptions the first time certain tasks access storage using the GETMAIN macroinstruction. The virtual machine assist extension for the IBM 3033 extension feature (6850) must be in place so that the user can get MVS/SP page fault assist on the IBM 3033 processor.

N

named saved system. Synonym for *saved system*.

NAMES file entry. In the MPVM NAMES or MPVMSYS NAMES file, a nickname tag/value pair and a set of related tag/value pairs that further describe the nickname.

network. (1) (TC97) An interconnected group of nodes. (2) Any set of two or more computers, work stations, or printers linked in such a way as to let data be transmitted between them.

network address. In SNA, an address, consisting of subarea and element fields, that identifies a link, a link station, or a network addressable unit. Subarea nodes use network addresses; peripheral nodes use local addresses. The boundary function in the subarea node to which a peripheral node is attached transforms local addresses to network addresses and

vice versa. See *local address*. See also *network name*.

network identifier (network ID). The network name defined to NCPs and hosts to indicate the name of the network in which they reside. It is unique across all communicating SNA networks.

network name. (1) In SNA, the symbolic identifier by which end users refer to a network addressable unit (NAU), a link, or a link station. See also *network address*. (2) In a multiple-domain network, the name of the APPL statement defining a VTAM application program is its network name and it must be unique across domains.

network node. Synonym for *node*.

network operator. (1) A person or program responsible for controlling the operation of all or part of a network. (2) The person or program that controls all the domains in a multiple-domain network.

Networking. (1) Accessing the CVIEW virtual machine through the IBM Pass-Through Facility. (2) In a multiple-domain network, communication among domains.

Networking session. A consult or conference session that includes one or more networking users.

Networking user. A session participant who accesses the CVIEW virtual machine through the IBM Pass-Through Facility.

node. (1) A single processor or a group of processors in a teleprocessing network. (2) A computer, work station, or printer, when it is participating in a network. (3) In SNA, an endpoint of a link or junction common to two or more links in a network. Nodes can be distributed to host processors, communication controllers, cluster controllers, or terminals. Nodes can vary in routing and other functional capabilities. (4) In VTAM, a point in a network defined by a symbolic name. Synonymous with *network node*. See *major node* and *minor node*.

node ID. Node identifier.

node identifier (node ID). The name by which a node is known to all other nodes in a network.

node name. In VTAM, the symbolic name assigned to a specific major or minor node during network definition.

noninteractive. The classification given to a virtual machine depending on the virtual machine's processing characteristics. When a virtual machine usually uses all its allocated queue slice, it is classified as being noninteractive or compute bound. Contrast with *interactive*.

noninteractive user. A user whose virtual machine completed its previous queue slice without having an I/O interrupt from the terminal.

nonresident pages. Pages whose contents are on DASD but not in real storage. A page is considered nonresident when an attempt to load its real address returns a nonzero condition code.

notepad. A facility that allows a user to get a copy of the current physical display in a file on his host user ID's A-disk.

Notify mode. The state in which session mode is suspended until CVIEW receives a response for each NOFIFY panel displayed in the session.

Notify panel. A panel that advises a user that one or more participants are sharing his interactive session.

NSS. Named saved system.

nucleus. The part of CP and CMS resident in main storage.

null line. A logical line with a length of zero that usually signals the CMS Editor to end input mode and enter edit mode. In VM/ESA, a null line for typewriter terminals is a terminal input line consisting of a return character as the first and only information, or a logical line end symbol as the last character in the data line. For display devices, a null line is indicated by the cursor positioned at the beginning of the user input area or the data in the user input area ending with a logical line end symbol.

O

object. (1) In computer security, a passive entity that contains or receives information. Access to an object implies access to the information it contains. Example of objects in VM/ESA are: minidisks, spool files, named saved segments, and virtual storage. Contrast with *subject*. (2) In VMSES/E, a usable form defined in build lists. (3) In VMSES/E, a built part of a product. A product consists of many objects, such as, nuclei, modules, execs, help files, and macrolibraries. (4) See *external object*, *object code*, *object module*, *public object*.

object code. Compiler or assembler output that is executable machine code or is suitable for more processing to produce executable machine code. Contrast with *source code*.

object module. A module that is the output of an assembler or a compiler and is input to a linkage editor.

offline. (1) Pertaining to the operation of a functional unit that takes place either independently of, or in parallel with, the main operation of a computer. (T)

(2) Neither controlled by, nor communicating with, a computer. Contrast with *online*.

OLTS. Online test system.

OLTSEP. Online test stand-alone executive program.

online. (1) Pertaining to the operation of a functional unit when under the direct control of the computer. (T) (2) Pertaining to a user's ability to interact with a computer. (A) (3) Pertaining to a user's access to a computer via a terminal. (A) (4) Controlled by, or communicating with, a computer. (5) Contrast with *offline*. (6) See *online message*, *online test stand-alone executive program (OLTSEP)*, *online test system (OLTS)*, *simultaneous peripheral operations online (SPOOL)*.

online message. The text that appears in the top area of a logon screen.

online test stand-alone executive program (OLTSEP). A program IBM uses for I/O maintenance.

online test system (OLTS). A system that lets a user test I/O devices concurrently with execution of programs. Tests can be run to diagnose I/O errors, verify repairs and engineering changes, or to periodically check devices.

operand. Information entered with a command name to define the data on which a command processor operates and to control the execution of the command processor.

Operating System/Virtual Storage (OS/VS). A family of operating systems that control IBM System/370 computing systems. OS/VS includes VS1, VS2, MVS/370, and MVS/XA.

Operating System/Virtual Storage (OS/VS). A family of operating systems that control System/370 computing systems. OS/VS includes VS1, VS2, MVS/370, MVS/XA, and MVS/ESA.

operator. A person who operates a machine. See *network operator*.

option. (1) A specification in a statement that may be used to influence the execution of the statement.

OS/VS. Operating System/Virtual Storage.

overhead. The additional processor time charged to each virtual machine for the CP functions needed to simulate the virtual machine environment and for paging and scheduling time.

override. (1) Synonym for *component parameter override*. (2) See *override area*, *override file*, *class override file*, *component override*, *component override area*, *product parameter override file*.

override $PPF. Synonym for *product parameter override file*.

override area. Synonym for *component override area*.

override file. Synonym for *class override file* and *product parameter override file*.

P

pack. A set of flat, circular recording surfaces that a disk storage device uses. Sometimes called a "disk pack."

page. (1) A fixed-length block that has a virtual address and can be transferred between real storage and auxiliary storage. (2) See *external page storage*, *external page storage management*, *inactive page*, *locked page*, *MVS page fault assist*, *nonresident pages*, *page-aligned*, *page frame*, *page frame table*, *page locking*, *page number*, *page reclamation*, *page table*, *page zero*, *pageable nucleus*, *pageable virtual machine*, *pseudo page fault*, *reserved page frame performance option*, *shadow page table*.

page frame. A block of 4096 bytes of real storage that holds a page of virtual storage.

page frame table. A table (called the CORTABLE) that contains an entry for each frame. Each frame table entry describes how the frame is being used.

page locking. Marking a page as nonpageable so that it remains in real storage until released.

page number. The part of a virtual storage address needed to refer to a page.

page reclamation. Making available and addressable the contents of a page in real storage that is waiting to be (or already has been) paged out. Page reclamation can occur after a page fault or after a request to fix or load a page, as long as no page-in operation has been scheduled for that particular page frame.

page table. A table (labeled PAGTABLE) that indicates whether a page is in real storage and that correlates virtual addresses with real storage addresses.

page zero. Storage locations 0 to 4095.

page-aligned. In fixed-block architecture, having a starting block number that is a multiple of eight and containing a number of blocks that is a multiple of eight.

pageable nucleus. Less frequently used portions of the CP nucleus not usually resident in real main storage.

pageable virtual machine. Synonymous with *virtual=virtual machine*.

paging. Transferring pages between real storage and external page storage.

panel. (1) A formatted display of information that appears on a terminal screen. See also *help panel* and *task panel*. Contrast with *screen*. (2) In computer graphics, a display image that defines the locations and characteristics of display fields on a display surface.

parameter. (1) A variable that is given a constant value for a specified application and that may denote the application. (2) See also *component parameter override*, *extended PLIST (untokenized parameter list)*, *parameter driven installation*, *parameter list (PLIST)*, *product parameter file (PPF)*, *product parameter override file*, *tokenized PLIST (parameter list)*, *untokenized parameter list*.

parameter driven installation. An installation method in which instructions for the installation are given at the beginning of the process rather than in response to prompts during the process.

parameter list (PLIST). In CMS, a string of 8-byte arguments that call a CMS command or function. The first argument must be the name of the command or function to be called. General register 1 points to the beginning of the parameter list.

part. (1) A CMS file provided on a product tape or service tape as input to the build process. A part is the smallest serviceable unit of a component. (2) See *part handler*, *parts catalog*, *PTF parts list*, *VM part catalog table*.

part handler. An exec provided by VMSES/E that builds a specific type of object or loads parts from service media.

Participant. Any user engaged in a CVIEW consult or conference session.

parts catalog. In VMSES/E, a set of software inventory files that catalog all parts of a product on a minidisk or SFS directory. All product parts are cataloged when they are loaded onto the system, when they are generated, and when they are moved.

password. In computer security, a string of characters known to the computer system and a user, who must specify it to gain full or limited access to a system and to the data stored within it.

patch. A circumventive service change applied directly to object code in a text deck in a nucleus.

path. (1) In APPC/VM or IUCV, a connection between two application programs that are on the same or different systems. Paths have names assigned to them. (2) In SNA, the series of path control network components (path control and data link control) that are traversed by the information exchanged between two network addressable units (NAUs). A path consists of a virtual route and its route extension, if any. See also *explicit route*. (3) In defining a switched major node, a potential dial-out port that can be used to reach a physical unit.

path control (PC) network. In SNA, the part of the SNA network that includes the data link control and path control layers. See *SNA network* and *user application network*.

PA1 key. The Program Attention key; used by a consultant or moderator to suspend session mode and access their control panel.

PC. Personal computer.

performance option. One or more functions that can be assigned to a virtual machine to improve its performance, response time (if terminal-oriented), or throughput under VM/ESA.

personal computer (PC). A desk-top, floor-standing, or portable microcomputer that usually consists of a system unit, a display monitor, a keyboard, one or more diskette drives, internal fixed-disk storage, and an optional printer.

PF key. Programmed function key.

physical saved segment. A member saved segment or discontiguous saved segment in which logical saved segments may be defined. See also *logical saved segment*, *saved segment*.

physical screen. Synonym for *screen*.

physical segment definition file. A file that identifies the logical saved segments to be included within a physical saved segment.

PLIST. Parameter list.

PLU. Primary logical unit.

polling. (1) * Interrogation of devices for purposes such as to avoid contention, to determine operational status, or to determine readiness to send or receive data. (2) (TC97) The process whereby stations are invited, one at a time, to transmit.

port table. An RSCS control block that describes the allocation status of switchable line ports available to RSCS.

PPF. Product parameter file.

preferred auxiliary file. In CMS, an auxiliary file that applies to a particular version of a source module to be updated, if multiple versions of the module exist.

preferred virtual machine. A particular virtual machine that has one or more of the performance options assigned to it.

prepared. In SFS, a synonym for *in-doubt*.

prerequisite. (1) A change that must be applied to the system before another change can be applied. (2) In VMSES/E, at the system level, a product that must be installed before another product can be installed. (3) In VMSES/E, at the service level, a PTF that must be applied before another PTF can be applied.

Presenter. A conference session participant who has an active keyboard.

preventive service. The application of all PTFs from a PUT or RSU. Contrast with *selective preventive service*.

primary link. In RSCS, one of a number of preferred links to use for transmitting data to a node. Contrast with *alternate link*.

primary logical unit (PLU). In SNA, the logical unit (LU) that contains the primary half-session for a particular LU-LU session. Each session must have a PLU and secondary LU (SLU). The PLU is the unit responsible for the bind and is the controlling LU for the session. A particular LU may contain both primary and secondary half-sessions for different active LU-LU sessions. Contrast with *secondary logical unit*.

primary paging device. The DASD with the highest effective data rate available for CP use. CP always attempts to store pages on this paging device instead of on slower paging devices if space on the primary paging device is available.

primary system operator privilege class. The CP privilege class A user. This operator has primary control over the VM/ESA system and can enable and disable teleprocessing lines, lock and unlock pages, force users off the VM/ESA system, issue warning messages, query, and set (and reset) performance options for selected virtual machines, and invoke VM/ESA accounting. If the current primary system operator logs off, the next class A user to log on becomes the primary system operator.

private gateway. (1) A gateway that programs outside a CS or TSAF collection can use to access private resources inside the collection. (2) A gateway that nonglobal resource manager programs can use to access resources outside a CS or TSAF collection. Contrast with *global gateway* and *system gateway*.

private resource. A resource accessible from anywhere within a CS or TSAF collection or SNA network and whose identity is known only within a single workstation or virtual machine. Contrast with *global resource*, *local resource*, and *system resource*.

privilege class. One or more classes assigned to a virtual machine user in a VM/ESA directory entry; each privilege class specified lets a user access a logical subset of the CP commands. There are nine IBM-defined privilege classes that correspond to specific administrative functions. They are:

Class A - primary system operator
Class B - system resource operator
Class C - system programmer
Class D - spooling operator
Class E - system analyst
Class F - service representative
Class G - general user
Class H - reserved for IBM use
Class Any - available to any user.

The privilege classes can be changed to meet the needs of an installation. See *class authority* and *user class restructure (UCR)*.

Privileged commands. CVIEW commands that control and administer the operation of the CVIEW virtual machine.

Privileged user.. A user, defined during customization, who is allowed to issue CVIEW privileged commands to the CVIEW virtual machine from his own virtual machine console.

problem determination. The process of identifying the source of a problem; for example, a program component, a machine failure, telecommunication facilities, user or contractor-installed programs or equipment, an environmental failure such as a power loss, or a user error.

Procedures Language VM/REXX. A component of VM/ESA. It contains the Procedures Language VM/REXX Interpreter, which processes the REXX language. It also contains the VM implementation of the SAA Procedures Language.

process. (1) A systematic sequence of operations to produce a specified result. A process is usually logical, not physical.

product. (1) Any separately installable software program, whether supplied by IBM or otherwise, distinct from others and recognizable by a unique identification code. The product identification code is unique to a given product, but does not identify the release level of that product. (2) See *product parameter file (PPF)*, *product parameter override file*, *software product*, *product service upgrade (PSU)*, *product tape*.

product parameter file (PPF). A file containing installation and service parameters for a product: control options, minidisk and SFS directory assignments, and component part type/function lists.

product parameter override file. A file containing one or more component override areas.

product service upgrade (PSU). A procedure used to upgrade the service level of a product or component using a recommended service upgrade (RSU) tape.

product tape. A tape containing components or products to load and build.

profile. A routine that defines *set-up* information upon entry into another routine.

PROFILE EXEC. A special EXEC procedure with a file name of PROFILE that a user can create. The procedure is usually executed immediately after CMS is loaded into a virtual machine (also known as IPL CMS).

program stack. Temporary storage for lines (or files) being exchanged by programs that execute under CMS. See *console stack*.

program state. A state associated with each partner in a conversation. This state defines the functions that a communication program can issue at a given time.

program status word (PSW). An area in storage used to indicate the order in which instructions are executed, and to hold and indicate the status of the computer system.

program temporary fix (PTF). Code changes needed to correct a problem reported in an APAR. The corrected code is included in later releases. A PTF contains one or more APAR fixes. For object-maintained parts that are changed, the PTF includes replacement parts. For source-maintained parts that are changed, the PTF includes update files and replacement parts. Each PTF is unique to a given release of a product. If the same problem occurs in multiple releases of a product, a separate PTF is defined for each release.

program update service. Receiving the contents of a PUT, applying all or some of the changes, and rebuilding the serviced parts. See *preventive service* and *selective preventive service*.

program update tape (PUT). A tape containing a customized collection of service tapes (preventive service) to match the products listed in a customer's ISD (IBM Software Distribution) profile. Each PUT contains cumulative service for the customer's products back to earlier release levels of the product still supported. The tape is distributed to authorized customers of the products at scheduled intervals or on request.

programmable operator facility. This CMS facility enables automatic filtering and routing of messages from a specified virtual machine (for example, the operator) to a logical operator virtual machine in a local distributed or mixed environment. It also permits installation defined actions to be automatically performed.

programmed function (PF) key. On a terminal, a key that can do various functions selected by the user or determined by an application program.

programmed symbols. A feature that allows definition of user-indicated shapes or symbols.

prompt. A displayed message that describes required input or gives operational information.

prompting. An interactive technique that lets the program guide the user in supplying information to a program. The program types or displays a request, question, message, or number, and the user enters the desired response. The process is repeated until all the necessary information is supplied.

protocol. (1) (CCITT/ITU) A specification for the format and relative timing of information exchanged between communicating parties. (2) (TC97) The set of rules governing the operation of functional units of a communication system that must be followed if communication is to be achieved. (3) In SNA, the meanings of, and the sequencing rules for, requests and responses used for managing the network, transferring data, and synchronizing the states of network components. (4) A set of rules for communication that are mutually understood and followed by two communicating stations or processes. The protocol specifies actions that can be taken by a station when it receives a transmission or detects an error condition.

PS/2. IBM Personal System 2.

pseudo page fault. A facility available with VM/VS handshaking that lets the VS1 virtual machine dispatch another task while waiting for a page-in request to be completed for some other task. Without this facility, the entire virtual machine would wait until the page request was satisfied, even if higher priority tasks were ready to execute.

PSS. Program support services.

PSU. Product service upgrade.

PSW. Program status word.

PTF. Program temporary fix.

PTF parts list. A file that identifies the file name and file type of each part required to install a PTF. A PTF parts list is shipped on the service tape for each PTF on the tape.

public network. A network established and operated by communication common carriers or

telecommunication Administrations for the specific purpose of providing circuit-switched, packet-switched, and leased-circuit services to the public. Contrast with *user-application network*.

public object. An object that all subjects can access but *only trusted subjects* can modify. Since all subjects are allowed to access them, public objects must contain innocuous data.

PUT. Program update tape.

PVM. VM/Pass-Through Facility.

PVMG. VM/Pass-Through Facility Gateway.

PVMG user ID. The user ID of the virtual machine in which VM/Pass-Through Facility Gateway is installed.

Q

queue. (1) A list constructed and maintained so that the next data element to be retrieved is the one stored first. (T) This method is characterized as first-in-first-out (FIFO). (2) A line or list of items waiting to be processed; for example, virtual machines waiting to run or messages to be displayed. (3) To arrange in or form a queue.

quiesce mode. The condition of the PVM server virtual machine when the SHUTDOWN command has been issued without the IMMED operand. This allows users' sessions to complete normally, but does not allow any new sessions to be established.

quiet recording mode. The mode of operation where transient processor or main storage errors corrected or circumvented by hardware RETRY or error correction code logic are not recorded on the VM/ESA error recording cylinders. This mode is entered by the SET MODE RETRY QUIET command or after 12 transient machine checks have occurred while in full-recording mode.

Quit PF key. The program function key which, when pressed in response to a menu or data-entry panel during setup, removes a user from setup mode (and, thus, from CVIEW). This key also terminates an entire CVIEW session when pressed in response to a panel in session mode.

R

R/W. Read/write.

raddr. The real device address of an I/O device.

read-only access. An access mode associated with a virtual disk or SFS directory that lets a user read, but not write or update, any file on the disk or SFS directory.

real address. The address of a location in real storage or the address of a real I/O device.

real system operator. Any user who loads and runs VM in the real machine. Contrast with *virtual machine operator*.

receive. (1) Bringing into the specified buffer data sent to the user's virtual machine from another virtual machine or from the user's own virtual machine. (2) To load service files from a service tape. (3) See *receive message log*, *receive history log*, *receive ID*, *receive status table*.

receive history log. A file, $VMFREC $HISTORY, generated by the VMFREC EXEC that lists the CMS files in each tape file loaded from the service tape.

receive ID. A 7- or 8-character alphanumeric identifier that is used to name the software inventory files created during receive processing.

receive message log. A file, in which the VMFREC EXEC writes status and error messages during receive processing.

receive status table. The software inventory table that contains the relationship between a product and the $PPF file used to install it. It also identifies what products of PTFs have been received or committed. The file type of the system level inventory table is SYSRECS and the file type of the service level inventory table is SRVRECS.

Recipient. The user, in a consult session, who receives help or assistance from a consultant. A recipient always has an active keyboard.

recovery machine. The first machine to join a virtual machine group. It has responsibility for executing routines that were set with the GCS MACHEXT macro and cleaning up system resources when machines leave the group.

recovery server. See *CRR recovery server*.

register. See *general register*.

release. For VTAM to relinquish control of resources (communication controllers or physical units). See also *resource takeover*. Contrast with *acquire*.

remote. (1) Two entities (for example, a user and a server) are said to be remote to each other if they belong to different systems within a collection, or to different nodes within an SNA network. (2) In RSCS, a connection through one or more RSCS-defined links. Remote describes computers, I/O devices, and virtual machines, to and from which information must pass over an RSCS-defined link. From the perspective of a particular node in an RSCS network, remote refers to any other node in the network. Contrast with *local*. (3) Synonym for *link-attached*.

remote entry services (RES). In OS/VS1, the set of functions added to the job entry subsystem that allows jobs and their associated data to be entered from remote devices, processed at the central system, and then transmitted back to remote devices.

remote operator console facility (ROCF). A 4300 Series Support Processor licensed internal code function that permits communication from a remote console for functions like IML or IPL using a switched line. The VM/Pass-Through Facility program provides a communication vehicle that lets any of its supported display stations serve as this remote console.

Remote Spooling Communications Subsystem Networking (RSCS). An IBM licensed program and special-purpose subsystem that supports the reception and transmission of messages, files, commands, and jobs over a computer network.

replacement part. Synonym for serviceable part.

replacement service. Servicing a part by replacing the part with a new one.

reply. (1) A response to an inquiry. (2) In SNA, a request unit sent only in reaction to a received request unit.

requester. (1) The name given to a virtual machine containing a user program that requests a resource. (2) The program that relays a request to another computer through the SRPI. Contrast with *server*.

RES. Remote entry services.

reserved file types. (1) File types recognized by the CMS editors (EDIT and XEDIT) as having specific default attributes that include record size, tab settings, truncation column, and uppercase or lowercase characters associated with that particular file type. The CMS Editor creates a file according to these attributes. (2) File types recognized by CMS commands; that is, commands that only search for and use particular file types or create one or more files with a particular file type.

reserved page frame performance option. A virtual machine option that reserves a specific number of page frames for one virtual machine. Generally, this option lets the most recently active pages of storage in a virtual machine remain allocated in real storage.

resource. A program, a data file, a specific set of files, a device, or any other entity or a set of entities that the user can uniquely identify for application program processing in a VM system.

resource ID. A one-to-eight character name that identifies a resource.

resource manager. An application running in a server virtual machine that directly controls one or more VM resources. There are four categories of VM resource managers: global, local, private, and system. Also, a resource manager (such as the SFS file pool server), may participate in CRR.

resource takeover. In VTAM, action initiated by a network operator to transfer control of resources from one domain to another. See also *acquire* and *release*. See *takeover*.

response time. (1) The time between the submission of an item of work to a computing system and the return of results. (2) In systems with time sharing, the time between the end of a block or line-end character of terminal input and the display of the first character of system response at the terminal.

restricted saved segment. A segment space or discontiguous saved segment that can be accessed by a virtual machine only if the directory entry for the virtual machine contains a NAMESAVE control statement that specifies the name of the segment space or discontiguous saved segment.

Restructured Extended Executor (REXX) language. A general-purpose programming language, particularly suitable for EXEC procedures, XEDIT macros, or programs for personal computing. Procedures, XEDIT macros, and programs written in this language can be interpreted by the REXX/VM interpreter. Contrast with *CMS EXEC language* and *EXEC 2 language*.

resynchronization. CRR function that is performed by the CRR recovery server when there has been a failure during sync point processing for a transaction. Resynchronization, which involves exchanging log names and comparing logical unit of work states, automatically attempts to complete the sync point process for the transaction. The goal of resynchronization is to maintain a consistent state (data integrity) among the protected resources involved in a transaction. Resynchronization may complete after the application ends. In very rare cases, such as an irrecoverable media failure or an operator error, resynchronization cannot complete and CRR lets operator intervention complete the transaction.

return code. A code used by a program, routine, or subroutine that influences the execution of succeeding instructions.

REXX language. Restructured Extended Executor language.

ROCF. Remote operator console facility.

route. A connection to another system by a logical link and one or more intermediate systems. In TSAF, many links and possible intermediate systems that allow the connection of one system to another.

route extension (REX). In SNA, the path control network components, including a peripheral link, that make up the portion of a path between a subarea node and a network addressable unit (NAU) in an adjacent peripheral node. See also *path*, *explicit route*, and *virtual route*.

router. An enhanced connectivity program that interprets requests for services and directs them to the applicable server. See *server-requester programming interface (SRPI)*.

routing group. A collection of nodes that are grouped together for ease of routing. Groups can be owned by other groups, thus allowing for the creation of extensive tree structures.

routing table. A CMS file that contains the information that controls the operation of the Programmable Operator Facility. It lets the Programmable Operator Facility recognize a message as a command, determine the action to take when a message comes in, and recognize the authorized users of programmable operator functions.

RSCS. Remote Spooling Communications Subsystem Networking.

RSCS exit facility. An interface between RSCS and exit routines consisting of exit points established by EXITCALL macros and your own exit routines.

S

SAA. Systems Application Architecture.

save area. An area of main storage in which contents of registers are saved.

saved segment. An area of virtual storage that is assigned a name and saved. Segment spaces, member saved segments, and discontiguous saved segments are defined by CP and saved in system data files. Logical saved segments are defined by CMS. A saved segment can be attached to and detached from a virtual machine and can be shared by many virtual machines. See *segment space*, *member saved segment*, *discontiguous saved segment*, and *logical saved segment*.

saved system. (1) The control program portion of an operating system that is assigned a name and saved in pageable format in a system data file. Loading an operating system into a virtual machine by specifying the name of a saved system is more efficient than loading it from a device number. (2) Synonymous with *named saved system*.

scale. A line on the XEDIT full-screen display, used for column reference.

screen. An illuminated display surface; for example, the display surface of a CRT. Synonymous with *physical screen*.

scroll. To move all or part of the display image vertically to display data that cannot be observed within a single display image. See also *page*.

SDLC. Synchronous data link control.

secondary logical unit (SLU). In SNA, the logical unit (LU) that contains the secondary half-session for a particular LU-LU session. An LU may contain secondary and primary half-sessions for different active LU-LU sessions. Contrast with *primary logical unit*.

secondary user. A user designated to receive a disconnected user's console messages and to enter commands to the disconnected user's console.

security category. One of the two components of a security label that specifies which area of information a subject is permitted to access or an object is permitted to contain. See also *security level*.

security level. One of the two components of a security label that specifies into which general class of sensitivity and confidentiality a subject or object falls. See also *security category*.

segment. (1) An architected segment or saved segment. (2) See *discontiguous saved segment (DCSS)*, *exclusive segment*, *logical saved segment*, *logical segment definition file*, *member saved segment*, *physical saved segment*, *physical segment definition file*, *restricted saved segment*, *segment interface*, *segment number*, *segment space*, *segment table*, *shared segment*, *system segment identification file*.

segment interface. A method of reading and writing records in which the application is responsible for assembling input segments into logical records and for disassembling logical output records into segments. The segment interface is used under the BSAM and QSAM access methods.

segment number. The part of a virtual storage address needed to refer to a segment.

segment space. (1) A saved segment that begins and ends on a megabyte boundary and contains 1 – 64 nonoverlapping member saved segments. A segment space is created by CP when member saved segments are defined. Access to a segment space provides access to all of its members. (2) See also *discontiguous saved segment*, *member saved segment*, *saved segment*.

segment table. In System/370 virtual storage systems, a table used in DAT to control user access to virtual storage segments. Each entry indicates the

length, location, and availability of a corresponding page table.

selective preventive service. The selective application of PTFs from a PUT or RSU. Contrast with *preventive service*.

separator. Synonym for *delimiter*.

server. (1) The general name for a virtual machine that provides a service for a requesting virtual machine. (2) The program that responds to a request from another computer or the same computer through SRPI. Contrast with *requester*. (3) See *communications server*, *CRR recovery server*, *file pool server machine*, *intermediate communications server*, *recovery server*, *server system*, *server-requester programming interface (SRPI)*, *trusted server*.

server system. A data processing system containing one or more servers providing services in response to a request from another computer.

server-requester programming interface (SRPI). (1) A protocol between requesters and servers in an enhanced connectivity network. Includes the protocol to define a cooperative processing subsystem. (2) The interface that enables enhanced connectivity between requesters and servers in a network.

service. (1) Changing a product after installation. (2) See *access method services (AMS)*, *callable services library (CSL)*, *circumventive service*, *console communication service (CCS)*, *corrective service*, *corrective service tape*, *local service*, *preventive service*, *program update service*, *replacement service*, *selective preventive service*, *serviceable part*, *service level*, *service-level inventory*, *service machine*, *service pool*, *service pool virtual machine*, *service representative privilege class*, *service routines*, *service tape*, *service virtual machine*, *serviceability enhancements support stage (SESS)*, *system service program (SSP)*, *system services control point (SSCP)*, *update service*, *VMSES/E installation/service tool*, *VTAM service machine (VSM)*.

service level. The PTF or preventive service level that is associated with the testing level and support level of an orderable product function.

service machine. A virtual machine running a program that provides system-wide services.

service pool. A set of service pool virtual machines, designed to do work with a CMR.

service pool virtual machine. A virtual machine that is one of a set of identical virtual machines, created through the use of the POOL directory statement.

service representative privilege class. The CP privilege class F user, usually an IBM Service Representative, is permitted to set or change the level of detail of I/O device error recording to the level desired. See *full recording mode*, *intensive recording mode*, and *quiet recording mode*.

service routines. CP or CMS routines used for addressing and updating directories; formatting or initializing disks; or doing disk, tape, or terminal I/O functions.

service tape. A tape containing service changes for one or more products. See *corrective service tape* and *program update tape (PUT)*.

service virtual machine. A virtual machine that provides a system service such as accounting, error recording, monitoring, or that provided by a supported licensed program.

service-level inventory. Synonym for *service level software inventory*.

service-level software inventory. (1) A file maintained by VMSES/E that shows: requisite relationships between PTFs; the status of the PTFs installed; the level of each part of the product serviced; and the status of objects built for the product. (2) See also *system-level software inventory*. (3) Synonymous with *service-level inventory*.

serviceability enhancements support stage (SESS). The component of VM that helps you install and apply service to VM.

serviceable part. (1) In VMSES/E, An individual part of a product that can be serviced separately. A serviceable part has the file name of the source or replacement part and a file type in the form *tttnnnnn*, where *ttt* is a unique three-character abbreviation for the part type and *nnnnn* is the PTF number. Serviceable parts are maintained by both source updates and replacement service. (2) Contrast with *usable part*. (3) Synonymous with *replacement part*.

Session. (1) The SNA term for a connection between two LUs. The LUs involved allocate conversations across sessions. (2) In SNA, a logical connection between two network addressable units (NAUs) that can be activated, tailored to provide various protocols, and deactivated, as requested. Each session is uniquely identified in a transmission header (TH) by a pair of network addresses, identifying the origin and destination NAUs of any transmissions exchanged during the session. See *LU-LU session*, *SSCP-LU session*, *SSCP-PU session*, and *SSCP-SSCP session*. See also *LU-LU session type*. (3) A consultation or a conference supporting at least two CVIEW users at separate display stations.

session activation request. In SNA, a request that activates a session between two network addressable units (NAUs) and specifies session parameters that

control various protocols during session activity; for example, BIND and ACTPU. Synonymous with *generic bind*. Contrast with *session deactivation request*.

session deactivation request. In SNA, a request that deactivates a session between two network addressable units (NAUs); for example, UNBIND and DACTPU. Synonymous with *generic unbind*. Contrast with *session activation request*.

session driver. An RSCS program that permits communication between RSCS and an SNA remote work station or system. Contrast with *line driver*.

Session mode. The state in which all participants view identical screen images and share a single virtual machine. CVIEW session mode begins when setup mode is complete and ends when one participant logs off the shared virtual machine.

session parameters. In SNA, the parameters that specify or constrain the protocols (such as bracket protocol and pacing) for a session between two network addressable units (NAUs). See also *logon mode*.

Session PF key. The program function key assigned to the SESSION function in the DXGKEYS LIST file. The moderator uses the SESSION PF key to start a conference session and to return to a conference in progress.

session-initiation request. In SNA, an initiate or logon request from a logical unit (LU) to a system services control point (SSCP) that an LU-LU session be activated.

Setup mode. The state during which CVIEW users define a session and their roles in the session. Setup mode begins when a user issues the DIAL CVIEW command and ends when the VM/SP logo appears on the participants' screens.

sever. (1) Ending communication with another virtual machine or with the user's own virtual machine.

SFS. Shared file system.

SFS communication adapter. The part of CMS in a user machine that communicates with file pool server machines.

SFS directory. A group of files. SFS directories can be arranged to form a hierarchy in which one directory can contain one or more subdirectories as well as files.

shadow page table. A page table that CP creates and uses to control the virtual storage of a virtual machine operating system (such as DOS/VS or OS/VS) that itself can perform paging on a real machine.

shared. Pertaining to the availability of a resource to more than one use at the same time.

shared file system (SFS). A part of CMS that lets users organize their files into groups known as *directories* and selectively share those files and directories with other users.

shared segment. One or more segments of real storage, saved in a saved system, member saved segment, or discontiguous saved segment, that can be shared among many virtual machines. For example, in a CMS saved system, the CMS nucleus is shared in real storage by all virtual machines that loaded CMS by name; that is, every CMS virtual machine maps a 1MB segment of virtual storage to the same 1MB of real storage.

signal. (1) An information packet that indicates the occurrence of an event. (2) To announce that an event has occurred.

simulated logon. A session-initiation request generated when a VTAM application program issues a SIMLOGON macroinstruction. The request specifies a logical unit (LU) with which the application program wants a session in which the requesting application program will act as the primary LU (PLU).

simultaneous peripheral operations online (SPOOL). (1) (Noun) An area of auxiliary storage defined to temporarily hold data during its transfer between peripheral equipment and the processor. (2) (Verb) To use auxiliary storage as a buffer storage to reduce processing delays when transferring data between peripheral equipment and the processing storage of a computer.

single user group. The concept in GCS of a virtual machine that runs applications that do not require group communications. This allows an application to run without the overhead of group initialization and multiple virtual machines. Multiple users can IPL the same saved system if it had been built for a single user environment. See *virtual machine group*.

sink virtual machine. In VMCF, the virtual machine that receives messages or data from a source virtual machine. Contrast with *source virtual machine*.

SIO. Start I/O.

SMSG function. A CP function that lets a virtual machine send a special message to another virtual machine programmed to accept and process the message. See also *special message*.

SNA. Systems Network Architecture.

SNA network. The part of a user-application network that conforms to the formats and protocols of Systems Network Architecture. It enables reliable transfer of data among end users and provides

protocols for controlling the resources of various network configurations. The SNA network consists of network addressable units (NAUs), boundary function components, and the path control network.

SNA terminal. A terminal that supports Systems Network Architecture protocols.

SNA/CCS terminal. Any terminal accessing VM that is managed by a VTAM service machine.

software product. Any software supplied by IBM or an Original Equipment Manufacturer (OEM), or user written programs. The term includes program offerings and program products (PPs).

source code. The input to a compiler or assembler, written in a source language. Contrast with *object code*.

source file. A file that contains source statements for such items as high-level language programs and data description specifications.

source update file. A file containing a single change to a statement in a source file. The file can also include requisite information for applying the change. Synonymous with *update file*.

source virtual machine. In VMCF, the virtual machine that initiates the sending of messages or data to another virtual machine. Contrast with *sink virtual machine*.

spanned record. A logical record that spans across one or more data blocks on a storage device.

special message. A data transmission, made up of instructions or commands, sent from one virtual machine to another by means of the SMSG function. A special message is processed by the receiving virtual machine and does not appear on the receiver's console. See also *SMSG function*.

SPM. Synchronization point manager

SPOOL. Simultaneous peripheral operations online.

spool. (1) (Noun) An area of auxiliary storage defined to temporarily hold data during its transfer between peripheral equipment and the processor. (2) (Verb) To use auxiliary storage as a buffer storage to reduce processing delays when transferring data between peripheral equipment and the processing storage of a computer.

spool file. A collection of data along with CCWs for processing on a unit record device. Contrast with *system data file*.

spool file block. A 4096-byte buffer that contains control information, in addition to records. Synonymous with *spool file buffer linkage block*.

spool file buffer linkage block. Synonym for *spool file block*.

spool file class. A one-character class associated with each virtual unit record device. For input spool files, the spool file class lets the user control which input spool files are read next; and, for output spool files, it lets the spooling operator better control or reorder the printing or punching of spool files having similar characteristics or priorities. The spool file class value can be A through Z or 0 through 9.

spooling. The processing of files created by or intended for virtual readers, punches, and printers. The spool files can be sent from one virtual device to another, from one virtual machine to another, and to real devices. See *virtual console spooling*.

spooling devices. I/O devices (card readers, punches, printers, DASD) that read input and write output.

spooling operator privilege class. The CP privilege class D user who controls the real unit record equipment and all closed spool files.

SR. Symptom record.

SRPI. Server-requester programming interface.

SSCP. System services control point.

SSCP-LU session. In SNA, a session between a system services control point (SSCP) and a logical unit (LU); the session enables the LU to request the SSCP to help initiate LU-LU sessions.

SSCP-PU session. In SNA, a session between a system services control point (SSCP) and a physical unit (PU); SSCP-PU sessions allow SSCPs to send requests to and receive status information from individual nodes in order to control the network configuration.

SSCP-SSCP session. In SNA, a session between the system services control point (SSCP) in one domain and the SSCP in another domain. An SSCP-SSCP session is used to initiate and terminate cross-domain LU-LU sessions.

SSP. System service program.

stack. See *console stack* and *program stack*.

state. See *program state*.

station. (1) One of the input or output points of a network that uses communication facilities; for example, the telephone set in the telephone system or the point where the business machine interfaces with the channel on a leased private line. (2) One or more computers, terminals, or devices at a particular location.

status area. The area in the bottom right corner of the screen that contains messages about the status of a virtual machine.

string. A group of minidisks defined for a specific function in the product parameter file, for example, the BASE2 string, which holds source code.

subcommand. The commands of processors such as EDIT or XEDIT that run under CMS.

subject. In computer security, an active entity that causes information to flow among objects or changes the system's state. A subject is generally a person, process, or device. Contrast *object*.

subsystem. A secondary or subordinate system, usually capable of operating independent of, or asynchronously with, a controlling system.

subtags. In an MPVM NAMES or MPVMSYS NAMES file, tags that further describe a session nickname.

supervisor. The part of a control program that coordinates the use of resources and maintains the flow of processing unit operations.

supervisor call (SVC). A request that serves as the interface into operating system functions, such as allocating storage. The SVC protects the operating system from inappropriate user entry. All operating system requests must be handled by SVCs.

supervisor call instruction (SVC). An instruction that interrupts a program being executed and passes control to the supervisor so that it can do a specific service indicated by the instruction.

SVC. (1) Supervisor call instruction. (2) With X.25 NPSI, switched virtual circuit.

switched line. A communication line in which the connection between the communication controller and a remote link station is established by dialing.

switched virtual circuit (SVC). An X.25 NPSI circuit that is dynamically established when needed. The X.25 equivalent of a switched line.

symbolic destination name. A name an APPC/VM or CPI-Communications connection uses. Symbolic destination names index SNA routing and security parameters, which are stored in communications directory NAMES files, to complete a connection.

synchronization point manager (SPM). The part of the Coordinated Resource Recovery (CRR) facility that resides in the application's virtual machine. A protected resource registers with the SPM to participate in CRR. The SPM coordinates the updating of protected resources and distributes the coordination of protected conversations to other SPMs.

synchronization point processing. Consists of the SPM driving the participating resource adapters through the following SPM exits:

- Precoordination - checks participating resources to ensure they are ready for a sync point.
- Coordination - is the actual sync point, which implements the one-phase and two-phase commit protocols.
- Postcoordination - performs cleanup processing after a sync point.

There are also the following exits, but they are not considered sync point exits:

- End of work unit - does cleanup processing before the work unit ends.
- Backout required - puts the protected resource in a state such that rollback (backout) is required.

synchronous communication. A method of communication in a conversation where a user's virtual machine is put in a WAIT state and cannot issue any communication functions on any conversations until a particular function completes.

synchronous data link control (SDLC). A data link over which communication is conducted using the synchronous data protocol.

synonym. In CMS, an alternative command name defined by the user as equivalent to an existing CMS command name. Synonyms are entries in a CMS file with a file type of SYNONYM. Entering the SYNONYM command allows use of those synonyms until that terminal session ends or until the use of synonyms is revoked by entering the SYNONYM command with no operands.

syntax. The rules for the construction of a command or program.

system administrator. The person responsible for maintaining a computer system.

system analyst privilege class. The CP privilege class E user, usually the VM/ESA system analyst, who can query, examine, and print or display, but not change, certain areas of the CP nucleus and can create saved systems.

system configuration file. A file that resides on the parm disk and contains information about how CP should initialize the system.

system data file (SDF). A file defined by CP to hold a collection of data associated with a particular system function, such as a saved segment, a saved system, a printer image library, user class restructure definitions, a message repository, or system trace data. Because a system data file contains no CCWs, it cannot be processed on a unit record device. Contrast with *spool file*.

system disk. In VMSES/E, a minidisk or SFS directory containing the other products that are required during service.

system gateway. A gateway that lets APPC programs access global or private resources on a specific system within a CS or TSAF collection. APPC programs in a TSAF collection can also use the system gateway to access resources from an adjacent CS collection. Similarly, an APPC program in a CS collection can access resources in an adjacent TSAF collection. Contrast with *global gateway* and *private gateway*.

System Product Editor. The CMS facility, comprising the XEDIT command and XEDIT subcommands and macros, that lets a user create, change, and manipulate CMS files.

System Product Interpreter. The language processor of the VM operating system that processes procedures, XEDIT macros, and programs written in the REXX language.

system profile. An EXEC (SYSPROF) that resides in a saved system or on a system disk and called by CMS initialization. It contains some initialization functions, and provides a means for installations to override the default CMS environment by tailoring the exec to suit the installation.

system programmer privilege class. The CP privilege class C user; usually, the VM/ESA system programmer, who can change the contents of any real storage locations in the machine.

system resource. A resource that is unique on the system on which it is defined (similar to a local resource) but is accessible (similar to a global resource) from remote systems in a CS or TSAF collection, or in the SNA network. Contrast with *local resource*, *global resource*, and *private resource*.

system resource operator privilege class. The CP privilege class B user, who controls all the real resources of the machine, such as real storage, disk drives, and tape drives, not controlled by the primary system or spooling operators.

system segment identification file. A file (SYSTEM SEGID) that identifies the logical saved segments on the system and the physical saved segments in which they reside.

system service program (SSP). In ACF/TCAM, an IBM-supplied or user-supplied program that does system-oriented auxiliary functions in support of the message control program. System service programs run under control of the initiator as attached subtasks.

system services control point (SSCP). In SNA, the focal point within an SNA network for managing the configuration, coordinating network operator and problem determination requests, and providing directory support and other session services for end users of the network.

system-level inventory. Synonym for *system-level software inventory*.

system-level software inventory. (1) A file maintained by VMSES/E that contains requisite relationships between products or components; mapping of product identifier to the name of the $PPF file used during installation; the status of the product or component on the system; and mapping of PTF file type abbreviation to real CMS file type. (2) See also *service-level software inventory*. (3) Synonymous with *system-level inventory*.

Systems Application Architecture (SAA). A defined set of interfaces, conventions, and protocols that can be used across various IBM systems.

Systems Network Architecture (SNA). The description of the logical structure, formats, protocols, and operational sequences for transmitting information units through and controlling the configuration and operation of networks.

T

T-disk. Synonym for *temporary disk*.

TAG. Tag queue element.

tag queue element (TAG). An RSCS control block describing the files enqueued for processing by RSCS. Also called *tag slot*.

tag slot. Synonym for *tag queue element*.

takeover. The process by which the failing active subsystem is released from its extended recovery facility (XRF) sessions with terminal users and replaced by an alternate subsystem. See *resource takeover*.

target. (1) One of many ways to identify a line to be searched for by XEDIT. A target can be specified as an absolute line number, a relative displacement from the current line, a line name, or a string expression.

target application. An application on a target system that a user is accessing through PVM.

task. A basic unit of work used for the execution of a program or a system function.

task ID. A 2-byte field that uniquely defines a task within a GCS virtual machine. Task ID is sometimes combined with machine ID to uniquely identify a task within a virtual machine group.

task name. A one- to four-character alphanumeric string the RSCS operator uses to identify a task when communicating with RSCS about that task.

task panel. Online display from which you communicate with the program in order to accomplish the program's function, either by selecting an option provided on the panel or by entering an explicit command. See *help panel*.

TCB. Trusted computing base.

telecommunication line. Any physical medium such as a wire or microwave beam, that is used to transmit data. Synonymous with *transmission line*.

temporary disconnect. Disconnecting from an PVM session with the intent that you will reconnect within a short period of time.

temporary disk. An area on a DASD available to the user for newly created or stored files until logoff, at which time the area is released. Temporary disk space is allocated to the user during logon or when entering the CP DEFINE command. Synonymous with *T-disk*.

terminal. (1) A device, usually equipped with a keyboard and a display, capable of sending and receiving information. (2) See *control terminal*, *control unit terminal (CUT)*, *display terminal*, *distributed function terminal (DFT)*, *logically connected terminal*, *SNA/CCS terminal*, *terminal input buffer*, *terminal session*, *terminal user*, *typewriter terminal*.

terminal input buffer. Holds lines entered at the user's terminal until CMS processes them.

terminal session. The time from logon to logoff when a user and the virtual machine can use the facilities of VM/ESA or the operating system or both. This also includes any time that the virtual machine is running in disconnect mode. See *disconnect mode*.

terminal user. Anyone who uses a terminal to log on to VM/ESA.

terminate. In SNA, a request unit that is sent by a logical unit (LU) to its system services control point (SSCP) to cause the SSCP to start a procedure to end one or more designated LU-LU sessions.

text library. A CMS file that contains relocatable object modules and a directory that indicates the location of each of these modules within the library.

threshold value. The file size or number of records in a file that is used by a transmission algorithm to determine the stream to which a file is assigned.

time out. A time interval allotted for certain operations to occur; for example, response to polling or addressing before system operation is interrupted and must be restarted.

time sharing. A method of using a computing system that lets many users execute programs concurrently and interact with the programs during execution.

time sharing option (TSO). An optional configuration of the operating system that provides conversational time sharing from remote stations.

time stamp. A record containing the TOD clock value stored in its internal 32-bit binary format.

time-of-day (TOD) clock. A hardware feature required by VM/ESA. The TOD clock is incremented once every microsecond, and provides a consistent measure of elapsed time suitable for the indication of date and time; it runs regardless of the processor state (running, wait, or stopped).

timeout. A condition where one of PVM's internal timers reached its maximum setting. One example of a timeout is when a user does not do any activity for 30 minutes and MAXIDLE is set to 30 minutes.

TOD clock. Time-of-day clock.

token. An eight-character symbol created by the CMS EXEC processor when it scans an EXEC procedure or EDIT macro statements. Symbols longer than eight characters are truncated to eight characters.

token ring network (TRN). A network that uses a ring topology, in which tokens are passed in the circuit from node to node. A node ready to send can capture the token and insert data for transmission.

tokenized PLIST (parameter list). A string of doubleword aligned parameters occupying successive doublewords.

TPN. Transaction program name.

trace table. Synonym for *CP trace table*.

transaction. See *logical unit of work* (in terms of CRR) or *LUWID*.

transaction program name (TPN). A symbolic name given to a particular transaction program in an SNA-defined network.

transmission line. Synonym for *telecommunication line*.

Transparent Services Access Facility (TSAF). A component of VM that handles communication between systems by letting APPC/VM paths span multiple VM systems. TSAF lets a source program connect to a target program by specifying a name that the target has made known, instead of specifying a user ID and node ID.

trusted computing base (TCB). In computer security, all of the protection mechanisms within a computer system, including hardware, software, and firmware, the combination of which enforces a security policy. It creates a basic protection environment and provides additional user services required for a trusted computer system.

trusted server. A machine that runs programs necessary to the system's operation. (These programs provide services such as accounting, error recording, security, network management, printing, and many others.) A trusted server always runs disconnected, and it never performs work on behalf of some other user. Though not all servers are privileged, most are able to issue CP privileged commands. A RACF virtual machine is an example of a trusted server.

TSAF. Transparent Services Access Facility.

TSAF virtual machine. The virtual machine that lets user programs connect to and communicate with virtual machines on different VM systems.

TSO. Time sharing option.

TUSER. A user field in the communications vector table (CVT).

U

UCR. User class restructure.

unbind. In SNA, a request to deactivate a session between two LUs. See also *session deactivation request*. Contrast with *bind*.

unit record device. A reader, a printer, or a punch.

universal class card reader. A virtual card reader that can read any class of reader, printer, or punch files spooled or transferred to it.

untokenized parameter list. A parameter list in which no restrictions are placed on the structure of the items in the list. See *token* and *tokenized PLIST (parameter list)*.

update file. Synonym for *source update file*.

update service. Servicing a part by applying a change to a source file statement, then assembling or compiling the source file to produce a new object file.

UPSI. User program switch indicator.

user. (1) Anyone who requests the services of a computing system. (2) See *authorized user ID*, *class A user*, *class Any user*, *class B user*, *class C user*, *class D user*, *class E user*, *class F user*, *class G user*, *CMS user disk*, *general user privilege class*, *installed user program (IUP)*, *inter-user communication vehicle (IUCV)*, *interactive user*, *memo-to-users*, *multiple user mode*, *noninteractive user*, *secondary user*, *single user group*, *terminal user*, *user class*, *user class restructure (UCR)*, *user data*, *user exit*, *user ID*, *user identification card*, *user input area*, *user memo*, *user modification*, *user profile table*, *user program*, *user program area*, *user program switch indicator (UPSI)*, *user-written CMS command*, *user-written stage command*.

user. Anyone who requests the services of a computing system.

user class. A privilege category assigned to a virtual machine user in the user's directory entry; each class specified allows access to a logical subset of all the CP commands. See *privilege class*.

user class restructure (UCR). The extension of the class structure of CP instructions from 8 to 32 classes for each user, command, and DIAGNOSE code within the system. This extension allows the installation greater flexibility in authorizing CP instructions.

user data. In a file pool, any data that resides in storage groups 2 through 32767.

user exit. An interface to VM/ESA that can be used by an application program. Generally, a user exit affects only the particular application specifying the exit and is run as part of the application program.

user ID. User identification.

user identification. A one- to eight-character alphanumeric symbol identifying each terminal user.

user identification card. See *ID card*.

user input area. On a display device, the lines of the screen where the user is required to enter command or data lines. See *display mode*, *input line*, and *line mode*.

user memo. (1) At the system-level, special instructions for installing a product. (2) At the service-level, special instructions for installing a PTF. (3) *memo-to-users*.

user modification. Any change that a user originates for a product or component. See also *local service*.

user profile table. A table of user attributes (called virtual machine control block (VMBLOK)) built for each active user from information gathered during logon.

user program. A transaction program that requests a service from a resource manager program. User programs reside in requester virtual machines.

user program area. In CMS, the virtual storage area occupying location X'20000' to the end of the user's virtual machine. The beginning of the user program

area is the default loading point for user programs and for many CMS commands.

user program switch indicator (UPSI). An operand of the CMS SET command. The user can set the switches (1 byte) to a desired value, which can be tested by a program in CMS/DOS.

user-application network. A configuration of data processing products, such as processors, controllers, and terminals, established and operated by users for the purpose of data processing or information exchange, which may use services offered by communication common carriers or telecommunication Administrations. Contrast with *public network*.

user-written CMS command. Any CMS file created by a user that has a file type of MODULE or EXEC. Such a file can be executed as if it were a CMS command by issuing its file name, followed by any operands or options expected by the program or EXEC procedure.

V

V=V machine. Virtual=virtual machine.

vaddr. Virtual address.

virtual address. The address of a location in virtual storage. A virtual address must be translated into a real address to process the data in processor storage.

virtual card reader. CP's simulation on disk of a real card reader. A virtual card reader can read card, punch, or print records of up to 151 characters in length. The virtual device type and I/O device address are usually defined in the VM/ESA directory. See *spool file class* and *universal class card reader*.

virtual console. A console simulated by CP on a terminal such as a 3270. The virtual device type and I/O address are defined in the VM/ESA directory entry for that virtual machine.

virtual console spooling. The writing of console I/O on disk as a printer spool file instead of, or in addition to, having it typed or displayed at the virtual machine console. The console data includes messages, responses, commands, and data from or to CP and the virtual machine operating system. The user can invoke or terminate console spooling at any time. When the console spool file is closed, it becomes a printer spool file. Synonymous with *console spooling*.

virtual disk. (1) A logical subdivision (or all) of a physical disk storage device that has its own address, consecutive storage space for data, and an index or description of the stored data so that the data can be accessed. (2) A virtual disk is also called a *minidisk*.

virtual interval timer assist. A hardware assist function, available only on a processor, that has ECPS. It provides, if desired, a hardware updating of each virtual machine's interval timer at location X'50'.

virtual machine (VM). (1) A functional equivalent of a computing system. In VM/ESA, virtual machines can simulate the System/370, 370-XA, ESA/370, and ESA/390 functions. In addition, on ESA/390 systems, the XC virtual machine architecture is available. Each virtual machine is controlled by an operating system. VM controls the concurrent execution of several virtual machines on an actual processor complex. (2) See *authorized virtual machine*, *AVS virtual machine*, *ESA virtual machine*, *expanded virtual machine assist*, *guest virtual machine (GVM)*, *in-queue virtual machine*, *multiple-access virtual machine*, *pageable virtual machine*, *preferred virtual machine*, *service pool virtual machine*, *service virtual machine*, *sink virtual machine*, *source virtual machine*, *TSAF virtual machine*, *virtual machine (VM)*, *virtual machine assist (VMA)*, *virtual machine communication facility (VMCF)*, *virtual machine control block (VMBLOK)*, *virtual machine group*, *virtual machine operator*, *virtual machine operator's console*, *virtual machine reset*, *virtual=virtual machine (V=V machine)*, *XA virtual machine*, *XC virtual machine*, *370 virtual machine*.

virtual machine assist (VMA). A hardware feature available on certain VM/ESA-supported System/370 models that causes a significant reduction in the real supervisor state time that VM/ESA uses to control the operation of virtual storage systems such as VSE, DOS/VS and OS/VS and, to a lesser extent, CMS, DOS, and OS when running under VM/ESA. VM/ESA supervisor state time is reduced because the VMA feature, instead of VM/ESA, intercepts and handles interruptions caused by SVCs, other than SVC 76, and certain privileged instructions. See *CP assist*, *expanded virtual machine assist*, *Extended Control Program Support (ECPS:VM/370)*, and *virtual interval timer assist*.

virtual machine communication facility (VMCF). A CP function that provides a method of communication and data transfer between virtual machines operating under the same VM/ESA system.

virtual machine control block (VMBLOK). The primary control block for many activities related to a single virtual machine. This block contains, for each virtual machine, the following types of information: the dispatch and priority level of the virtual machine, the virtual machine's processor registers, preferred virtual machine options currently in effect, and information concerning all other significant activities.

virtual machine group. The concept in GCS of two or more virtual machines associated with each other through the same named system (for example, IPL GCS1). Virtual machines in a group share common

read/write storage and can communicate with one another through facilities provided by GCS. Synonymous with *group*. See *single user group*.

virtual machine operator. Any user who loads and runs an operating system in a virtual machine. Contrast with *real system operator*.

virtual machine operator's console. The virtual machine operator's logon terminal, which simulates the hardware system console functions for the guest operating system. Messages destined for the hardware system console are presented to the virtual machine operator's console.

virtual machine reset. A collective term for the simulation of the system-reset-normal or system-reset-clear processes defined by the System/370 or System/390 hardware architectures. Among other things, virtual machine reset is an event that causes CP to recover many resources associated with a virtual machine, such as IUCV or APPC paths and data spaces. Virtual machine reset can result from issuing such CP commands as SYSTEM CLEAR, SYSTEM RESET, IPL or LOGOFF.

Virtual Machine Serviceability Enhancements Staged/Extended (VMSES/E). A component of VM/ESA that provides the tools for installing and servicing the various components of the VM product after VM/ESA Rel. 1.0. It is also the strategic installation and service tool for all of the other products that run on those VM/ESA platforms.

Virtual Machine/Enterprise Systems Architecture (VM/ESA). An IBM licensed program that manages the resources of a single computer so that multiple computing systems appear to exist. Each virtual machine is the functional equivalent of a *real* machine.

Virtual Machine/System Product (VM/SP). An IBM licensed program that manages the resources of a single computer so that multiple computing systems appear to exist. Each virtual machine is the functional equivalent of a *real* machine.

Virtual Machine/VTAM Communications Network Application (VM/VCNA). A program that runs in the VTAM service machine. VM/VCNA controls the physical appearance of the screen when displaying output on a VM/ESA terminal attached to an SNA network.

virtual printer. A printer simulated on disk by CP for a virtual machine. The virtual device type and I/O address are usually defined in the VM/ESA directory entry for that virtual machine.

virtual route (VR). In SNA, a logical connection (1) between two subarea nodes that is physically realized as a particular explicit route, or (2) that is contained wholly within a subarea node for intra-node sessions. A virtual route between distinct subarea nodes imposes a transmission priority on the underlying explicit route, provides flow control through virtual-route pacing, and provides data integrity through sequence numbering of path information units (PIUs). See also *explicit route*, *path*, and *route extension*.

virtual spooling device. Unit record devices simulated on disk by CP. The virtual device type and I/O addresses are defined in the VM/ESA directory or else by the CP DEFINE command. CP's spooling facilities let multiple virtual machines share the real unit record devices. In addition, VM/ESA has extended the concept of spooling to include the logging of all I/O to and from the virtual machine system console. See *virtual card reader, virtual console spooling,* and *virtual printer (or punch)*.

virtual storage. Storage space that can be regarded as addressable main storage by the user of a computer system in which virtual addresses are mapped into real addresses. The size of virtual storage is limited by the addressing scheme of the computing system and by the amount of auxiliary storage available, not by the actual number of main storage locations.

virtual storage extended (VSE). The generalized term that indicates the combination of the DOS/VSE system control program and the VSE/Advanced Functions licensed program. Note that in certain cases, the term DOS is still used as a generic term; for example, disk packs initialized for use with VSE or any predecessor DOS or DOS/VS system are sometimes called DOS disks. Also note that the DOS-like simulation environment provided under the VM/ESA CMS component and CMS/DOS exists on VM/ESA licensed programs and continues to be called CMS/DOS.

Virtual Telecommunications Access Method (VTAM). An IBM licensed program that controls communication and the flow of data in a computer network. It provides single-domain, multiple-domain, and multiple-network capability. VTAM runs under MVS, OS/VS1, VM/ESA, and VSE.

virtual=fixed machine (V=F machine). A preferred virtual machine with a fixed, contiguous area of host real storage that does not start at page 0. CP provides performance enhancements for this virtual machine. See also *multiple preferred guests*, *preferred virtual machine*, *virtual=real machine*.

virtual=real area (V=R area). The part of real storage, starting with real page 1, where a virtual=real machine can execute. CP maintains control of real page zero; only page zero of the virtual=real machine is relocated. Only one virtual machine at a time can occupy the virtual=real area. The area must be defined to contain the largest virtual=real machine likely to run.

virtual=real machine (V=R machine). A preferred virtual machine with a fixed, contiguous area of host real storage that starts at page 0. CP provides performance enhancements and an automatic recovery facility for this virtual machine. See also *multiple preferred guests*, *preferred virtual machine*, *virtual=real area*, *virtual=real machine recovery*, and *virtual=virtual machine*.

virtual=real machine recovery (V=R machine recovery). A CP function that lets the V=R machine resume operation after most CP abnormal terminations. When possible, the facility reestablishes the V=R machine environment, allowing the operating system running in that virtual machine to perform its own recovery processes. See *automatic software re-IPL*.

virtual=virtual machine (V=V machine). A virtual machine that runs in the dynamic paging area. CP pages this virtual machine's guest real storage in and out of host real storage. See *dynamic paging area*, *virtual=fixed machine*, and *virtual=real machine*.

VM. Virtual machine.

VM directory. A CP disk file that defines each virtual machine's typical configuration: the user ID, password, regular and maximum allowable virtual storage, CP command privilege class or classes allowed, dispatching priority, logical editing symbols to be used, account number, and CP options desired. Synonymous with *CP directory*.

VM part catalog table. A table that identifies the product or component that owns all files residing on a disk and the VMSES/E command that last modified or created the file. It is modified each time VMSES/E modifies or creates a file on a disk identified in the :MDA section of the product parameter file.

VM/ESA. Virtual Machine/Enterprise Systems Architecture.

VM/Pass-Through Facility. A facility that lets VM users interactively access remote system and processor nodes. These can be remote IBM 4300 processors, other VM systems, with or without this facility installed, or System/370- compatible non-VM systems.

VM/Pass-Through Facility Gateway. The VM/Pass-Through Facility component that provides a gateway to the SNA network.

VM/Pass-Through Facility Multiple Session Application. The multisession component of VM/Pass-Through Facility. This component is also known as MPVM.

VM/SP. Virtual Machine/System Product.

VM/VCNA. Virtual Machine/VTAM Communications Network Application.

VMA. Virtual machine assist.

VMBLOK. Virtual machine control block.

VMCF. Virtual machine communication facility.

VMSES/E. (1) Virtual Machine Serviceability Enhancements Staged/Extended. (2) See *VMSES/E installation/service tool*.

VMSES/E installation/service tool. Two VMSES/E user interfaces, VMFINS and VMFSIM, all of the VMSES/E commands, and the service-level and system-level software inventories. Usually referred to as *VMSES/E*.

VSE. Virtual storage extended.

VSM. VTAM service machine.

VTAM. Virtual Telecommunications Access Method.

VTAM definition. The process of defining the user application network to VTAM and modifying IBM-defined characteristics to suit the needs of the user.

VTAM operator. A person or program authorized to issue VTAM operator commands.

VTAM service machine (VSM). A virtual machine that contains an operating system (OS/VS1 or DOS/VSE), an access method (ACF/VTAM or ACF/VTAME), and VM/VCNA. VSM forms the interface for SNA communication in VM/ESA.

W

WAN. Wide area network.

wide area network (WAN). A network that provides communication services to a geographic area larger than that served by an LAN. Contrast with *local area network (LAN)*.

working set. The estimated number of pages of real storage that the virtual machine needs to execute.

workstation. An I/O device from which jobs can be submitted to a host system for processing, or to which output can be returned, or both.

wrap. The continuation of an operation from the maximum addressable location in storage to the first addressable location.

X

XA virtual machine. A virtual machine that simulates System/370-Extended Architecture functions. Contrast with *370 virtual machine*, *ESA virtual machine*, and *XC virtual machine*.

XC virtual machine. A virtual machine operating mode in which the virtual machine is simulating ESA/XC architecture. XC virtual machines can be provided only when VM/ESA is running on an ESA/390 processor. Contrast with *370 virtual machine, ESA virtual machine,* and *XA virtual machine*.

XEDIT. The CMS facility, containing the XEDIT command and XEDIT subcommands and macros, that lets a user create, change, and manipulate CMS files.

Z

zap. To modify or dump an individual text file, using the ZAP command or the ZAPTEXT EXEC.

270X. Refers to the IBM 2701, 2702, and 2703 Transmission Control Units or the Integrated Communications Adapter (ICA) on the System/370 Model 135.

3088. Refers to the IBM 3088 Multisystem Channel Communications Unit, Models 1 and 2.

3262. Refers to the IBM 3262 Printer, Models 1 and 11.

3270. Refers to a series of IBM display devices, for example, the IBM 3275, 3276 Controller Display Station; 3277, 3278, and 3279 Display Stations; the 3290 Information Panel; and the 3287 and 3286 printers. 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.

3270 data stream. All data transmitted through a channel in a single read or write operation to a terminal.

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.

3289. Refers to the IBM 3289 Model 4 Printer.

3310. Refers to the IBM 3310 Direct Access Storage Device.

3330. Refers to the IBM 3330 Disk Storage Device.

3340. Refers to the IBM 3340 Direct Access Storage Device.

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.

3375. Refers to the IBM 3375 Direct Access Storage Device.

3380. Refers to the IBM 3380 Direct Access Storage Device.

3390. Refers to the IBM 3390 Direct Access Storage Device.

370 mode. A GCS mode of operation on ESA that simulates 370 architecture.

370 virtual machine. A virtual machine that simulates System/370 functions. Contrast with *XA virtual machine, ESA virtual machine,* and *XC virtual machine*.

370x. Refers to the IBM 3704/3705 Communication Controllers.

3725. Refers to the IBM 3725 Communication Controllers.

3800. Refers to the IBM 3800 Printing Subsystems. A specific device type is used only when a distinction is required between device types.

4224. Refers to the IBM 4224 Printer.

4250. Refers to the IBM 4250 Printer.

9370. Refers to a series of processors, namely the IBM 9371 Models 10, 12, and 14, the IBM 9373 Model 20, the IBM 9375 Models 40 and 60, the IBM 9377 Model 90, and other models.

Bibliography

This bibliography lists the books that provide additional information about your system.

VM/Pass-Through Facility Library

The following table lists, by task-oriented category, the VM/Pass-Through Facility library books and their order numbers. Books cited in this document are marked with a box (■).

	Title	Order Number
Evaluation		
	VM/Pass-Through Facility: Licensed Program Specifications	GC24-5591
Administration, Operation and Service		
	VM/Pass-Through Facility: Administration and Operation	SC24-5557
End Use		
■	VM/Pass-Through Facility: User's Guide	SC24-5555
Diagnosis and Reference		
■	VM/Pass-Through Facility: Messages	SC24-5648
■	VM/Pass-Through Facility: Programming Reference	SC24-5556
■	VM/Pass-Through Facility: Auto-Signon Reference	SC24-5656

RSCS Library

The following table lists, by task-oriented category, the RSCS library books and their order numbers. Books cited in this document are marked with a box (■).

	Title	Order Number
Evaluation		
	RSCS Licensed Program Specifications	GH24-5223
	RSCS General Information	GH24-5218
Planning, Installation and Service		
	RSCS Planning and Installation	SH24-5219
End Use		
	RSCS Operation and Use	SH24-5220
	RSCS Reference Summary	SX24-5257
Diagnosis and Reference		
	RSCS Diagnosis Reference	LY24-5248
	RSCS Messages and Codes	SH24-5221
Miscellaneous		
	RSCS Data Interchange Manager	SH24-5248
	RSCS Exit Customization	SH24-5222
	RSCS 3270 Secure Printing	SH24-5255

© Copyright IBM Corp. 1991,1993

VM/ESA Release 2.1 Library

The following table lists, by task-oriented category, each VM/ESA Release 2.1 book and its order number. Books cited in this document are marked with a box (■).

	Title	Order Number
Evaluation		
	VM/ESA: Licensed Program Specifications	GC24-5439
	VM/ESA: General Information	GC24-5550
	VM/ESA: Introduction and Features Summary	SC24-5651
Installation and Service		
■	VM/ESA: VMSES/E Introduction and Reference	SC24-5444
	VM/ESA: Installation Guide	SC24-5526
■	VM/ESA: Service Guide	SC24-5527
Planning and Administration		
	VM/ESA: REXX/EXEC Migration Tool for VM/ESA	GC24-5607
■	VM/ESA: Connectivity Planning, Administration, and Operation	SC24-5448
■	VM/ESA: Planning and Administration	SC24-5521
	VM/ESA: Running Guest Operating Systems	SC24-5522
	VM/ESA: Conversion Guide and Notebook for VM/XA SP and VM/ESA	SC24-5525
	VM/ESA: Group Control System	SC24-5531
	VM/ESA: Performance	SC24-5642
	VM/ESA: SFS and CRR Planning, Administration, and Operation	SC24-5649
	VM/ESA: Conversion Guide and Notebook for VM/SP, VM/SP HPO, and VM/ESA 370 Feature	SC24-5654
Operation		
	VM/ESA: Virtual Machine Operation	SC24-5523
	VM/ESA: System Operation	SC24-5528
Application Programming		
	VM/ESA: CMS Application Development Guide	SC24-5450
	VM/ESA: CMS Application Development Reference	SC24-5451
	VM/ESA: CMS Application Development Guide for Assembler	SC24-5452
	VM/ESA: CMS Application Development Reference for Assembler	SC24-5453
	VM/ESA: CMS Application Migration Guide	SC24-5454
	VM/ESA: Programmer's Guide to the Server-Requester Programming Interface for VM	SC24-5455
	VM/ESA: REXX/VM User's Guide	SC24-5465
	VM/ESA: REXX/VM Reference	SC24-5466
■	VM/ESA: CP Programming Services	SC24-5520
	VM/ESA: Enterprise Systems Architecture/Extended Configuration Principles of Operation	SC24-5594
	VM/ESA: Common Programming Interface Communications User's Guide	SC24-5595
	VM/ESA: REXX/VM Primer	SC24-5598
	VM/ESA: CMS Application Multitasking	SC24-5652
	SAA Common Programming Interface Communications Reference	SC26-4399
	External Security Interface (RACROUTE) Macro Reference for MVS and VM	GC28-1366
	SAA Common Programming Interface Resource Recovery Reference	SC31-6821
End Use		
	VM/ESA: Master Index and Glossary	GC24-5518

	Title	Order Number
	VM/ESA: CMS Primer	SC24-5458
■	VM/ESA: CMS User's Guide	SC24-5460
■	VM/ESA: CMS Command Reference	SC24-5461
	VM/ESA: XEDIT User's Guide	SC24-5463
	VM/ESA: XEDIT Command and Macro Reference	SC24-5464
■	VM/ESA: CP Command and Utility Reference	SC24-5519
	VM/ESA: CMS Pipelines Reference	SC24-5592
	VM/ESA: CMS Pipelines User's Guide	SC24-5609
Diagnosis		
	VM/ESA: CMS Diagnosis Reference	LY24-5244
■	VM/ESA: CMS Data Areas and Control Blocks	LY24-5245
	VM/ESA: Diagnosis Guide	LY24-5250
	VM/ESA: CP Diagnosis Reference	LY24-5251
	VM/ESA: CP Data Areas and Control Blocks	LY24-5253
	VM/ESA: System Messages and Codes	SC24-5529
■	VM/ESA: Dump Viewing Facility	SC24-5530
Reference Summaries		
	VM/ESA: CP Diagnosis Reference Summary	LX24-5260
	VM/ESA: CMS Primer Summary of Commands	SX24-5247
	VM/ESA: CMS Command Reference Summary	SX24-5249
	VM/ESA: XEDIT Command Reference Summary	SX24-5250
	VM/ESA: REXX/VM Reference Summary	SX24-5251
	VM/ESA: Quick Reference	SX24-5259
	VM/ESA: CMS Pipelines Reference Summary	SX24-5270
	VM/ESA: CP Command and Utility Reference Summary	SX24-5271

Bill-of-Forms

To receive a set of books, use one of the following order numbers:

SBOF-3286	One set of unlicensed VM/ESA Release 2.1 books
LBOF-3287	One set of VM/ESA Release 2.1 books to help you diagnose system problems
LBOF-3288	One set of licensed VM/ESA Release 2.1 books

Related Books

The following table lists other books, outside the VM/Pass-Through Facility and VM/ESA libraries, that are helpful when using this book. Books cited in this document are marked with a box (■).

	Title	Order Number
VM/ESA 370 Feature		
■	VM/ESA: Installation for 370	SC24-5431
■	VM/ESA: Service Guide for 370	SC24-5429
■	VM/ESA: CP Planning and Administration for 370	SC24-5430
■	VM/ESA: CP System Command Reference for 370	SC24-5434
■	VM/ESA: Group Control System Reference	SC24-5426
■	VM/ESA: CP Programming Services for 370	SC24-5435

	Title	Order Number
■	VM/ESA: CP General User Command Reference for 370	SC24-5433
■	VM/ESA: Interactive Problem Control System Guide and Reference for 370	SC24-5438
■	VM/ESA CMS Application Development Guide for Assembler	SC24-5452
VM/XA System Product		
■	VM/XA SP: Dump Viewing Facility Operation Guide and Reference	SC23-0359
■	VM/XA SP: CP Programming Services	SC23-0370
■	VM/XA SP: CMS Application Program Development Reference	SC23-0370
■	VM/XA SP: Product Installation and Service	SC23-0364
Customer Information Control System/VM		
	CICS/VM: Problem Determination	SC33-0593
Other		
	Dictionary of Computing	SC20-1699

Index

Special Characters

?CONFIG nickname entry 98
?DEFAULT nickname entry 98
?FINAL nickname entry 99
?INITIAL nickname entry 98
* (comment) command 286
* (comment) record
  in a configuration file
    description 347
    placement in the configuration file 69
    use in the configuration file 70, 89
  in a message repository 240
* comment record, PVMG

Numerics

3088 line driver
  defining 71
    DEFINE command (PVM) 293
    LINK record 358
  description of 6
  removing function from PVM 201
  restrictions in a VM/ESA or VM/XA environment 37
3270 IDS printers
  restrictions 35
  sessions 115
3271/3274 emulation restrictions 34
327x line driver
  See also emulator lines
  adding devices to a 72
  defining 71
    DEFINE command (PVM) 293
    LINK record 358
  description of 5
  removing function from PVM 201
370 virtual machine mode 37
37XX emulation program restrictions, in a VM/ESA or VM/XA environment 37
4300 remote processor session 437

A

abnormal termination
  security considerations 32
accessing systems
  Non-PVM 5
  PVM 5, 7
  Remote IBM 4300 Processor 5
adding
  devices to a line 72
  directory entry for PVM 41
  links
    DEFINE command (PVM) 293
    LINK record 358
addresses for control units 23
administrative considerations 21
  enhancements to PVM in Version 2 21
  performance 29
  requirements 22
  restrictions 32
  supported devices 23
  system security 29
  VM/ESA and VM/XA environment 37
after initialization exit (DVMCUEAI) 222
after read exit (DVMCUEAR) 223
after write exit (DVMCUEAW) 224
alternate operator commands 109
  * (comment) 286
  AUDIT command, PVM 286
  AUTHORIZ, PVMG 378
  CDBC command, PVM 289
  CMD command, PVM 290
  CMS command, PVM 290
  CP command, PVM 291
  CPQ command, PVM 292
  DEFINE 109, 293
  DROP 113, 299
  EXEC, PVM 301
  LMSG 111, 302
  LMSG (PVMG) 118
  LOGMSG (PVM) 303
  MSG 111, 308
  MSG(PVMG) 119
  PAUSE command, PVM 313
  PMSG 111, 119, 314
  QUERY 107
  QUIESCE 112, 321
  RESUME 112, 323
  ROUTE command, PVM 324
  SELECT command, PVM 326
  SHOW 110, 328
  START command, PVM 330
  STATUS 332
  summary 284, 377
  TRACE command,PVM 334
  VARY 337
alternate route
  defining 74, 370
  sample use in the configuration file 74
AMSTAMP records, in a message repository 242
APPC/VM line driver
  defining 71, 417
    CMS communications directory 80
    DEFINE command (PVM) 293
    LINK record 358
  description 6
  removing function from PVM 202
  restrictions in a VM/ESA or VM/XA environment 37

© Copyright IBM Corp. 1991,1993

APPL statement 53
Application Major Node Table (for VTAM) 53
application programming 251—257
  command response interface 252
  programmable operator 251
  PVMPROP/CRI message protocol 252
APPLN record, PVMG 402
applying
  corrective service 147
  exits to PVM 225
  service automatically 168
  service to PVM Standalone 144
  service to PVM with VMSERV 143
  service using VMSES/E 141
ATTACH statements 42
AUDIT command, PVM 286
audit file 77
  AUDIT command, PVM 286
  AUDIT record 347
AUTHORIZ
  command, PVM 287
  command, PVMG 378
  record, PVMG 403
authorizing PVM operators 75
authorizing PVMG operators 91
AUTOLOG1 user ID 51
autologging virtual machines 51
automatic node selection 76
automating PVM initialization
  PROFILE EXEC for CP and CMS commands 79
  PVM profile exec for PVM commands 81
auxiliary storage
  requirements 26

B

backup language
  changing 235
  description of 228
base text staging minidisk 45, 229
before final exit (DVMCUEBF) 225
before initialization exit (DVMCUEBI) 222
before read exit (DVMCUEBR) 223
before write exit (DVMCUEBW) 224
bibliography 509
bill-of-forms 511
bringing up PVM 47
BSC connections 5
BSCA line driver
  defining 71
    DEFINE command (PVM) 293
    LINK record 358
  description of 5
  removing function from PVM 201

C

calculating values for
  BUFFERS record 350
calculating values for *(continued)*
  EXT record 354
  I/O record 356
  SET RESERVE value 200
CCWs restriction, logical device support 35
CDBC command, PVM 289
CDBC command, PVMG 380
changing the backup language 235
CINIT session initiation data exit (DVMNUE6) 216
class authority
  class A 38
  class B 22
  class E 183
  for CPTRAP use 183
  for IPCSSCAN use 183
  for MSGNOH use 365, 410
  installation considerations 42
   PVMG installation considerations 43
  when configuration file contains a MSGNOH record 22
  when configuration file contains CLPORT records 22
  when lines or devices attached by ATTACH or NET ATTACH 22
  when SET RESERVE is used 38
class, privilege
  class A 38
  class B 22
  class E 183
  for CPTRAP use 183
  for IPCSSCAN use 183
  for MSGNOH use 365, 410
  installation considerations 42
   PVMG installation considerations 43
  when configuration file contains a MSGNOH record 22
  when configuration file contains CLPORT records 22
  when lines or devices attached by ATTACH or NET ATTACH 22
  when SET RESERVE is used 38
cluster statistics exit (DVMNMEC) 218, 219
CMD command, PVM 290
  CMD command, PVM 290
CMD command, PVMG 381
  CMD command 381
CMS command, PVM 290
CMS communications directory 80
  using with APPC/VM links 80, 417
CMS GLOBALV command 312
CMS subset
  CMS command, PVM 290
  identifying CMS subset commands to PVM 75
  SUBSET configuration file record 372
  using subset commands 111
CMS SYNONYM file 312
CMS user exits 221—225
  after initialization (DVMCUEAI) 222

CMS user exits *(continued)*
after read (DVMCUEAR) 223
after write (DVMCUEAW) 224
before final (DVMCUEBF) 225
before initialization (DVMCUEBI) 222
before read (DVMCUEBR) 223
before write (DVMCUEBW) 224
COMBLOCK (IPCS) command 340
command and response exit (DVMNUE5) 215
command prefix
example 256
format 255
use in application programming 252
command response
for application programs
command response interface 252
deciphering encoded messages 253
PVMPROP/CRI message protocol 252
specifying a language for 255
command summary 284, 377
commands
CP commands used with PVM 338—339
CVIEW commands 124
descriptions 286—420
in CMS subset mode 111
invoking PVM under CMS 309
MPVM command 304
operator examples 103—117
prefixes 255
PVM commands 286
PVM/IPCS commands 339—420
specifying a language for responses 255
summary 284, 377
comment (*) command 286
Comment record, PVMG 401
commenting
audit file 286
configuration file 70, 89, 347
message repository 240
communications directory
using with APPC/VM links 80, 417
CONFIG nickname entry 98
connection types 5—6
considerations
auxiliary requirements 25
performance 29, 199—202
PVM in a VM/ESA or VM/XA environment 37
routing 74
time out 77
TRACE command 180
console
requirements for ROCF sessions 36
continuation character 241
control blocks
DVMAAT 340, 419
DVMACT 420
DVMALT 343, 420
DVMAPT 422
control blocks *(continued)*
DVMAUT 423
DVMBUF 425
DVMCOMN 427
DVMCPOR 429
DVMPORT 430
DVMSYSB 345, 431
control unit
defining addresses for emulator lines 23
support requirements on non-PVM systems 22
copy function restrictions 34
corrective service
applying 147
applying automatically 168
CORRSERV EXEC
automatically applying service 168
format 175
processing 173
CP command, PVM 291
CP commands used for PVM performance
DIAG98 directory option 199
SET FAVORED 199
SET QDROP OFF 199
SET RESERVE command 200
VCUNOSHR directory option 199
CP DIAL command
description 338
restrictions in a VM/ESA or VM/XA environment 37
CP directory
entry for PVM operator 41
options
DIAG98 199
VCUNOSHR 199
samples
MAINT user ID 44
PVM user ID 42
PVMG user ID 43
CP parameter on DUMP record 354
CP remote restrictions in a VM/ESA or VM/XA environment 37
CP SET command 312
CP SMSG command 339, 399
CPQ command, PVM 292
CPTRAP file 183
creating
audit file
AUDIT command, PVM 286
AUDIT record 77, 347
configuration file 66
message repository 238
problem report 189
CTCA line driver
defining 71
DEFINE command (PVM) 293
LINK record 358
description of 5
removing function from PVM 202

CTCA line driver *(continued)*
restrictions 34
customizing PVM
applying exits to PVM 225
changing the backup language 235
CMS user exits 221
enhancing performance 199
installing languages 228
network management exits 218
programmable operator 251
removing functions 201
session control exits 212
using a message repository 238
CVIEW
accessing the CVIEW virtual machine 13
attendee 462
conference session setup
ATTEND panel 462
CONFWAIT panel 463
moderator 464
PASSWORD panel 463
consult session setup
CONSPASS panel 461
consult session setup (consultant) 460
GIVE panel 460
GIVEWAIT panel 462
IDGIVE 460
REDISP panel 461
control mode panels
consultant control 467
moderator control 468
control-mode panels 467
customizing CVIEW 265
changing the PK keys 266
concurrent users 267
CONSULT IDNAME 265
defining listed consultants 265
DXGINIT exec 265
listed consultants 265
NOTIFY panel 267
PF key functions 266
privileged users 268
to accept a default 268
to change a default 268
to run DXGINIT 268
CVIEW commands 124
CMS command, CVIEW 124
CP command, CVIEW 124
DROP command, CVIEW 125
HELP command, CVIEW 126
PASSWORD command, CVIEW 126
QUERY command, CVIEW 127
SHUTDOWN command, CVIEW 129
STOP command, CVIEW 130
device type and model assignment 136
different display station types 17
directory entry 58
display-input option 16
CVIEW *(continued)*
DXGINIT EXEC 267
establishing a CVIEW session
conference session 131
consult session 131
establishing a session 130
installation planning 57
joining a conference 136
log file 137
machine requirements 58
managing a CVIEW session 133
conference sessions 134
consult sessions 133
messages 17
modes of operation 10
control mode 13
notify mode 12
session mode 11
setup mode 10
networking considerations 15
networking sessions 14
notify-mode 465
operating environment 58
PASSTHRU considerations 16
product overview 9
product panels 457
HOME panel 457
setup-mode panels 457
PROFILE EXEC 137
setup-mode
consult setup (recipient) 458
IDRECV panel 458
RECEIVE panel 458
RECWAIT panel 459
starting CVIEW 123
automatically 123
manually 123
technical overview 13
using CVIEW 123
virtual machine requirements 57
block multiplexer (BMX) 57
channel mode 57
disk space 57
nonshared virtual control units 57
privilege class 57
user ID 57
VCUNOSHR option 57
vitual storage 57

D

DASD requirements 25
data length restrictions
3271/3274 34
data length restrictions 33
ROCF 36
default
routing 74

DEFAULT nickname entry 98
DEFINE command
  description, PVM 293
  sample 109—110
defining
  alternate routes 74, 370
  line groups 73, 355
  lines 358
  links
    DEFINE command (PVM) 293
    LINK record 358
    permanent 71, 358
    temporary 109, 293
  message repository 243
  message severity characters 241
  ports for a 327x line driver 367
  ports for an R3270 line driver 351
  PVM configuration 63
    creating a configuration file 66
    planning 63
    sample configuration file 66
  PVM menus 243
  R3270 line driver 353
  time stamps 242
developing exits 209—226
  applying to PVM 225
  caution note 207
  CMS user 221—225
  network management 218—221
  session control 212—218
    sample 217
  time interval between calls to link statistics exit 372
device support 23
  in a VM/ESA or VM/XA environment 38
DEVQUE (IPCS) command 340
DIAG98 directory option 199
DIAL command 338
directory
  entry for PVM operator 41
  options
    DIAG98 199
    VCUNOSHR 199
  samples
    MAINT user ID 44
    PVM user ID 42
    PVMG user ID 43
DISC command, PVM 298
DISC command, PVMG 382
  DISC command 382
disconnecting the PVM operator's console 298
disconnecting the PVMG operator's console 382
dispatching priority 43, 201
display station support 23
  addressing 23
  in a VM/ESA or VM/XA environment 38
  valid types 367
displaying
  DVMAAT control block 190, 340
  DVMALT control block 190, 343
  DVMAUT control block 190, 346
  DVMCOMN control block 190, 340
  DVMSYSB control block 190, 345
  DVMTCB control block 190, 346
  PVM route table 190, 345
distributed data processing (DDP) 2
distribution tape
  layout 277
double-byte character restriction 374
DROP command, PVM 114, 117
  description 299
  DROP command, PVM 299
  samples 113
DROP command, PVMG 120, 382
  DROP command 382
  samples 120
dropping lines, sessions, and tasks (PVM DROP command) 299
dropping lines, sessions, and tasks (PVMG DROP command) 382
dump
  destination of 353
  DUMP record 77
  restrictions in a VM/ESA or VM/XA environment 37
  specifying type of 353
dump facilities 187—188
  full dumps 188
  partial dumps 188
  restrictions in a VM/ESA or VM/XA environment 37
DUMP record, PVM
  description 353
  example 68
  placement in the configuration file 69
  use in the configuration file 77
  using to create a problem report 189
DUMP record, PVMG 403
  DUMP record, PVMG 403
  use in the configuration file 91
DVMAAT control block 340, 419
DVMACT control block 420
DVMALT control block 343, 420
DVMAPT control block 422
DVMAUT control block 423
DVMBUF (Internal Buffer Header) 425
DVMCOMN control block 427
DVMCPOR control block 429
DVMCUEAI (after initialization exit) 222
DVMCUEAR (after read exit) 223
DVMCUEAW (after write exit) 224
DVMCUEBF (before final exit) 225
DVMCUEBI (before initialization exit) 222
DVMCUEBR (before read exit) 223

DVMCUEBW (before write exit) 224
DVMENTR macro 211
DVMNMEC (cluster statistics exit) 218, 219
DVMNMEL (link statistics exit) 218, 219
  setting time interval between calls 372
  time interval between calls 77
DVMNMEL (link statistics exit), time interval between calls 77
DVMNMES (session statistics exit) 219, 221
DVMNUE1 (session initiation exit) 212
DVMNUE2 (session termination exit) 213
DVMNUE3 (origin node session request exit) 214
DVMNUE4 (intermediate node session request exit) 214
DVMNUE5 (command and response exit) 215
DVMNUE6 (CINIT session initiation data exit) 216
DVMNUE6 (session initiation exit) 216
DVMPORT (Simulator Link Port Table Entry) 430
DVMRETN macro 211
DVMSYSB control block 345, 431
DVMXTRN macro 212

E

EFGDCSS exec 203
EFGTRACE exec 192
EFGZAP exec 204
emulator line driver
  See also emulator lines
  adding devices to a 72
  time interval between line status checks 373
emulator lines
  abnormal termination handling 32
  communicating by 5
  control unit addresses 23
  description 5
  restrictions 33
  time out handling 78
encoded messages
  deciphering 253
  placement in a command response message header 252
end highlighting character 242
enhancements to PVM Version 2 21
enhancing PVM performance
  DIAG98 directory option 199
  SET FAVORED 199
  SET QDROP OFF 199
  SET RESERVE command 200
  VCUNOSHR directory option 199
entering commands in CMS subset mode 111
entering CP commands within PVM (CP command) 291
ERASE ALL UNPROTECTED command restrictions 33
ERRLOG files 193
ESA virtual machine mode 37
ETRACE system trace 183
examining a dump 190
examples
  BUFFERS record calculation 350
  CMS communications directory 80
  configuration file 66—69
  EXT record calculation 354
  I/O record calculation 356
  message repository 248
  operator commands 103—117
  PROFILE EXEC 79
  PROFILE PVM 82—83
  session control routine 217
  SET RESERVE calculation 200
excluding double-byte character messages 71, 89
EXEC command, PVM 301
exits 209—226
  applying to PVM 225
  caution note 207
  CMS user 221—225
  network management 218—221
  session control 212—218
    sample 217
  time interval between calls to link statistics exit 372
extended data stream support 24

F

facilities for application programming 251—257
  command response interface 252
  programmable operator 251
  PVMPROP/CRI message protocol 252
feature tape
  installing 228
  languages included on 227
  layout 279
files
  See also audit file
  CMS communications manager 80
  CMS SYNONYM 312
  configuration file
    See PVM configuration file
  CPTRAP 183
  created by the message compiler 316
  distribution tape 277
  PROFILE EXEC
    for general users 312
    for the PVM user ID 79
  PROFILE PVM 81, 82—83
  PVMCONF.DAT 68
  service tape 278
  system directory
    for MAINT user ID 44
    for PVM user ID 42
    for PVMG user ID 43
  UCOMDIR NAMES 80
FINAL nickname entry 99

G

general use commands
  * (comment) 286
  CMD command, PVM 290
  command prefixes 255
  CP commands
    DIAL 338
    SMSG 339, 399
  CPQ command, PVM 292
  MSG 308
  PASSTHRU 309
  PMSG (PVM) 314
  QUERY 318
  SHOW 328
  summary 284, 377
getting national languages on your system 235
groups
  DEFINE command (PVM) 293
  GROUP record 355

H

hardware restrictions, in a VM/ESA or VM/XA environment 38
HELP (IPCS) command 341
HELP minidisk
  storage requirements 25
  used during installation 45
HELP online 284

I

identifying
  CMS subset commands 75
  local node name 71
  system language 70
  unique resource name 71
incorrect 3270 orders 34
INIT command, PVMG 383
  INIT command 383
Initial Menu
  bypassing 360
  defining PF key node settings 371
INITIAL nickname entry 98
installing PVM 55
  adding a directory entry for the PVM operator 41
  administrative considerations 21
  considerations
    performance 199
    routing 74
    storage 26
    time out 77
    TRACE command 180
  dispatching priority 43
  distribution tape files
    quick method 47, 48
    standalone method 47, 48
    tape layout 277
installing PVM *(continued)*
  exits 209—226
    See also developing exits
  feature tape files 228—235
    tape layout 279
  loading tape files 47
  MAINT user ID 44
  minidisks used
    base text staging 45
    HELP 45
    PVM system 45
    service staging 45
    source staging 45
    system extension 45
  performance options
    DIAG98 directory option 199
    dispatching priority 201
    removing functions from PVM 201
    ROCF sessions 201
    SET FAVORED 199
    SET QDROP OFF 199
    SET RESERVE 200
    tailoring the PVM MODULE 201
    VCUNOSHR directory option 199
  planning 21
  PVMG system directory 43
  requirements 22
  restrictions 33
    3271/3274 emulation 34
    Logical Device Support 35
    ROCF 36
    with VM/ESA and VM/XA 37
  service tape files 142—146
    tape layout 278
  storage considerations 26
  system directory 42
  verifying 54
  VM/XA considerations 38
installing PVMG
  adding a directory entry for the PVMG operator 43
installing the national language feature tape 228—235
  enabling MPVM for additional languages 233
  enabling PASSTHRU (DVMUSI) for additional languages 233
  enabling PVM for additional languages 233
  enabling PVMG for additional languages 233
  loading files 230
  MAINT minidisks used 229
  minidisks used 228
  verifying installation 233
INSTFPP EXEC 48
interactive PVM sessions
  samples
    with remote 4300 processor 437
  security considerations 31

intermediate node
  description 5
  in determining a routing path 74
  sample configuration file 68
intermediate node session request exit (DVMNUE4) 214
invoking MPVM 304
IPCS 147, 153, 160
  analyzing information with 189, 195
  commands
    COMBLOCK 340
    DEVQUE 340
    HELP 341
    IPCSDUMP 341
    IPCSSCAN 342
    LINKS 343
    MAP 344
    ROUTES 345
    SYSBLOCK 345
    TASKQUE 346
    USERQUE 346
  creating problem reports 189
  examining a dump 190
IPCSDUMP (IPCS) command
  creating a problem report 189
  description 341
IPCSSCAN (IPCS) command
  description 342
  examining dumps 190
  subcommands 190
    COMBLOCK 340
    DEVQUE 340
    LINKS 343
    ROUTES 345
    SYSBLOCK 345
    TASKQUE 346
    USERQUE 346
  using 183
IUCV directory statement 42, 43
IUCV paths, specifying maximum for PVM 362

L

L. command prefix 255
LANG record, PVMG 404
language
  backup language 228
  changing the backup language 235
  default national 357
  feature tape 227
  identifying the system language 70
  installing 228
  national 227, 235
  specifying
    with command prefixes 255
line
  327x restrictions 35
  addressing 81
  DEFINE command (PVM) 293
line *(continued)*
  defining 358
  defining for a 327x 367
  defining groups 355
  DROP command, PVM 299
  performance considerations 200
  QUIESCE command 322
  RESUME command 323
  ROCF restrictions 33
  routing 74
  START command, PVM 330
  types 5
  VARY command 337
line driver
  data transfer 182
  defining 71
    DEFINE command (PVM) 293
    LINK record 358
    ports for an R3270 351
    remote 3270 clusters 353
  removing from PVM 201
  restrictions in a VM/ESA or VM/XA environment 37
  status checks 77
  time out handling 78
  tracing
    I/O activity 181
  types 5–6
    3088
    327x
    APPC
    BSCA
    CTCA
    R3270
    ROCF
line groups
  defining 73
    DEFINE command (PVM) 293
    GROUP record 355
lines, tracing activity 181
link
  327x restrictions 35
  addressing 81
  DEFINE command (PVM) 293
  defining 358
  defining for a 327x 367
  defining groups 355
  DROP command, PVM 299
  QUIESCE command 322
  RESUME command 323
  ROCF restrictions 33
  routing 74
  START command, PVM 330
  types 5
  VARY command 337
LINK record, PVMG 404
link statistics exit (DVMNMEL) 218, 219
  setting time interval between calls 372

link statistics exit (DVMNMEL) *(continued)*
time interval between calls 77
LINKS (IPCS) command 343
LMSG command, PVM
description 302
LMSG command,PVM 302
sample 111
sample (PVMG) 118
LMSG command, PVMG 384
LMGS command 384
load map
use with MAP (IPCS) command 344
loading files
distribution tape 47
national language feature tape 230—233
service tape 142—146
local PVM system name 361
LOCAL record
LOCAL record, PVMG 406
SESSDISC record, PVMG 413
use in the PVMG configuration file 90
LOCAL record, PVMG 406
logging messages and screen images 193
Logical Device Support Facility 5
logical devices
restrictions 35
LOGMODE record, PVMG 406
LOGMSG command 303
LOGMSG command, PVM
LOGMSG command, PVM 303
LOGMSG command, PVMG 385
logo 132
logon logo 132
Logon Mode Table 54
cview.Multiple CVIEW virtual machines 59
LOGONID record, PVMG 407
LU names 53
LU record, PVMG 408
LU statement, PVMG 53

M

M. command prefix 255
MAINT user ID
DASD requirements 25
minidisks used by PVM
base text staging 45
base text staging minidisk 229
for national language HELP files 229
HELP 45
PVM system 45
PVM system disk 229
service staging 45
source staging 45
source staging minidisk 229
system extension 45
system extension minidisk 229
minidisks used during feature tape installation 229
MAINT user ID *(continued)*
storage requirements 25
used for applying service 142
used for installing PVM 41
MAP (IPCS) command 344
master record, in a message repository 241
memo-to-users
distribution tape 48
location on the distribution tape 277
location on the service update tape 278
message commands
CP SMSG 339, 399
LMSG 111
LMSG (PVM) 302
LMSG (PVMG) 118
LOGMSG 303
MSG 111
MSG (PVM) 308
MSG(PVMG) 118
PMSG 111
PMSG (PVM) 314
PMSG(PVMG) 118
message compiler
changing the backup language 235
compiling the message repository 238
description 316
files created by 316
message record (message repository) 243
message repository
AMSTAMP record 242
checking the accuracy of the message file 238
comment record 240
compiling the message file 238
continuation character 241
creating 238
creating a message file 238
defining message severity characters 241
end highlighting character 242
making available to users 239
master record 241
message records 243
PMSTAMP record 242
samples 248
screen records 243
special considerations 246
start highlighting character 241
start variable character 241
substitution variables 245
time stamp records 242
using 238
message severity characters 241
messages
defining in a message repository 243
LMSG command, PVM 302
LOGMSG command, PVM 303
MSG command, PVM 308
PMSG command, PVM 314
specifying a destination for operator messages 366

messages *(continued)*
specifying format of 365
messages, identifying 193
messages, logging 193
minidisk
base text staging 45
HELP 45
PVM system 45
service staging 45
source staging 45
system extension 45
used during distribution tape installation 41, 45
used during distribution tape PVMG installation 43
used during feature tape installation 229
used during service tape installation 143
used for national language HELP files 229
MODETAB parameter (on APPL statement) 54
modifying links, DEFINE command (PVM) 293
MPVM
command 304
command, return codes from 306
NAMES file 95
MPVMBLD command 307
MPVMSYS NAMES file 95
MSG command, PVM
description 308
MSG command, PVM 308
sample 111
sample(PVMG) 118
MSG command, PVMG 385
MSGNOH record, PVMG 410
MSGOPTNS record, PVMG 410

N

NAMES file 80, 95, 312
national language
backup language 228
changing the backup language 235
default national 357
feature tape 227
identifying the system language 70
installing 228
national 227, 235
specifying
with command prefixes 255
national language feature tape 279
installing 228
languages included on 227
layout 279
NETWORK command restrictions, in a VM/ESA or VM/XA environment 37
network line driver, time interval between line status checks 373
network management exits 218—221
cluster statistics (DVMNMEC) 218, 219
link statistics (DVMNMEL) 218, 219
session statistics (DVMNMES) 219, 221
networking terms 4
non-PUT service 142
non-PVM node, security considerations 30
notepad function
PASSTHRU command description 310
restrictions 33
security considerations 29

O

online HELP 284
operating PVM 103—117
operator
commands and execs
See also alternate operator commands
DISC command, PVM 298
QUIT command, PVM 323
RUNPVM command 325
RUNPVMG command 392
SHUTDOWN command, PVM 329
summary 284, 377
directory entry for 41
tasks 103—117
OPID record, PVMG 411
optional minidisks 45
origin node session request exit (DVMNUE3) 214
OUR-APPLID statement 53
override variable settings for current session 306

P

PA key
PASSTHRU command
description 309
setting defaults 312
path
defining 324, 370
displaying 110
ROUTE command, PVM 324
ROUTE record 370
SHOW command, PVM 328
PAUSE command, PVM 313
PCCF support task
defining 358
querying 318
removing function from PVM 202
restrictions 34
sample LINK record for 71
sample PVMCONF.DAT file 68
starting 330
terminating 299
tracing 182, 334
performance
commands
DIAG98 directory option 199
SET FAVORED 199
SET QDROP OFF 199
SET RESERVE 200
VCUNOSHR directory option 199

performance *(continued)*
considerations 29, 199—202
dispatching priority 43, 201
removing functions from PVM 201
ROCF sessions 201
tailoring the PVM MODULE 201
requirements with VM/ESA (ESA Feature) and VM/XA 38
PF keys
general user tasks
starting a PVM session 312
operator tasks
adding to PVM menus 326
SELECT command, PVM 326
settings for PVM menus 76, 371
planning your PVM configuration 63
PMSG command, PVM
description 314
PMSG command, PVM 314
sample 111
sample(PVMG) 118
PMSG command, PVMG 386
PMSG command, PVMG 386
PMSTAMP records, in a message repository 242
port
for a 327x line driver 367
for R3270 line driver 351
security considerations 31
types 367
prefix 255
prerequisite programming 22
preventative service
applying to PVM 142
description of 141
printer sessions
starting 115
stopping 117
types 115
printers
attaching 42
devices support 24
DROP command,PVM 299
QUERY command 319
QUIESCE command 322
restrictions 35
RESUME command 324
START command, PVM 330
support in a VM/ESA or VM/XA environment 38
types 35
privilege class
class A 38
class B 22
class E 183
for CPTRAP use 183
for IPCSSCAN use 183
for MSGNOH use 365, 410
installation considerations 42
PVMG installation considerations 43
privilege class *(continued)*
when configuration file contains a MSGNOH record 22
when configuration file contains CLPORT records 22
when lines or devices attached by ATTACH or NET ATTACH 22
when SET RESERVE is used 38
problem determination 179—190
analyzing information with IPCS 189, 195
creating problem reports with IPCS 189
examining dumps with IPCS 190
gathering information about your installation 179
TRACE facility 180
PROFILE PVM 81—83
program temporary fix (PTF)
applying corrective service 147
described 141
program update service
loading service tape files 142—146
program update tape (PUT)
described 141
loading tape files 142—146
programmable operator
description 251
messages sent to a 252
PVMPROP record 369
PVMPROP/CRI message protocol 252
specifying user ID for 369
programming interfaces xv
PTF
applying corrective service 147
described 141
PUT
described 141
loading tape files 142—146
PVM
accessible system types 5—6
accessing 7
connection types 5—6
DASD requirements 25
extended data stream support 24
in a VM/ESA and VM/XA environment 37
interactive sessions
line driver types 5—6
loading tape files 47
minidisks used
base text staging 45
HELP 45
PVM system 45
service staging 45
source staging 45
system extension 45
networking terms 4
operating 103—117
optional minidisks 45
printer sessions
starting 115
stopping 117

PVM *(continued)*
printer support 24
privilege class 22
required minidisks 45
restrictions 32—36
3270 IDS printers 35
3271/3274 emulation 34
CTCA 34
data length 33
Logical Device Support 35
Notepad function 33
ROCF 36
SYS REQUEST key 33
TEST REQUEST key 33
routing considerations 74
security considerations 29—32
abnormal termination 32
eliminating exposures 30
notepad function 29
PORT record 31
restricting port usage 31
session end function 30
temporary disconnect function 30
user session security 30
storage requirements 25
supported devices 23
system directory 42—43
time out considerations 77
PVM AUDIT record
description 347
example 68
placement in the configuration file 69
use in the configuration file 77
PVM AUTHORIZ record
description 349
example 68
placement in the configuration file 69
use in the configuration file 75
PVM AUTO record
description 349
placement in the configuration file 69
PVM BUFFERS record
calculating value for 350
description 350
placement in the configuration file 69
use in the configuration file 79
PVM CLPORT record
description 351
example 35, 67
placement in the configuration file 69
use in the configuration file 72
PVM CLUSTER record
description 353
example 67
placement in the configuration file 69
use in the configuration file 72
PVM commands 286—420
See also commands
PVM commands *(continued)*
prefixes 255
summary 284, 377
PVM comment (*) record
in a configuration file
description 347
placement in the configuration file 69
use in the configuration file 70, 89
in a message repository 240
PVM configuration file
adding devices to a line 72
327x line 72
R3270 line 72
authorizing operators 75
automatically connecting users to a node 76
commenting 347
commenting in 70, 89
creating 66
defining a link 71
defining alternate routes 74
defining line groups 73
excluding double-byte character messages 71, 89
identifying a unique resource name 71
identifying CMS subset commands 75
identifying the local node name 71
identifying the system language 70
order of records 69
records 347—374
sample 66
saving PVM information, audit file 77
saving PVM information, dumps 77
setting PF keys for PVM menus 76
specifying message requirements 78, 91
specifying storage requirements 79
specifying time restrictions 77
PVM EXT record
calculating entry size 354
description 354
placement in the configuration file 69
use in the configuration file 79
PVM GROUP record
description 355
example 67
placement in the configuration file 69
use in the configuration file 73
PVM I/O record
calculating entry size 356
description 356
placement in the configuration file 69
use in the configuration file 79
PVM interactive sessions
samples
with remote 4300 processor 437
security considerations 31
PVM LANG record
description 357
example 67
placement in the configuration file 69

PVM LANG record *(continued)*
use in the configuration file 70
PVM LINK record
description 358
example 67
placement in the configuration file 69
time out conditions 77
use in the configuration file 71
used to determine routing 74
PVM LNODE record
description 360
example 68
placement in the configuration file 69
use in the configuration file 76
PVM LOCAL record
description 361
example 67
placement in the configuration file 69
use in the configuration file 71
PVM MAXCONN record
description 362
placement in the configuration file 69
PVM MAXIDLE record
description of 362
PVM MAXSESS record
description of 363
PVM MAXSUSP record
description 363
PVM MAXTDISC record
description of 364
PVM MAXUSERS record
description of 364
PVM message compiler
changing the backup language 235
compiling the message repository 238
description 316
files created by 316
PVM messages
defining in a message repository 243
LMSG command, PVM 302
LOGMSG command, PVM 303
MSG command, PVM 308
PMSG command, PVM 314
specifying a destination for operator messages 366
specifying format of 365
PVM MSGNOH record
description 365
example 68
placement in the configuration file 69
use in the configuration file 78, 91
PVM MSGOPTNS record
description 365
example 68
placement in the configuration file 69
use in the configuration file 78, 91
PVM OPID record
description 366
PVM OPID record *(continued)*
placement in the configuration file 69
PVM PORT record
description 367
example 67
placement in the configuration file 69
restricted usage 368
use in the configuration file 72
used to prevent unauthorized access 31
PVM printer sessions
starting 115
stopping 117
types 115
PVM profile exec 81—83
PVM PVMPROP record
description 369
placement in the configuration file 69
PVMPROP/CRI message protocol 252
use in application programming 251
PVM route table, displaying 190
PVM SESSTERM record
description of 371
placement in the configuration file 69
PVM SET command
description of 326
PVM STATTIME record
description of 372
example 68
placement in the configuration file 69
use in the configuration file 77
PVM SUBSET record
description of 372
placement in the configuration file 69
use in the configuration file 75
PVM system disk 229
PVM system minidisk 45
PVM TDISC record
description 373
example 68
placement in the configuration file 69
use in the configuration file 77
PVM TIMEOUT record
description 373
example 68
placement in the configuration file 69
use in the configuration file 77
PVM XDBC record
description 374
example 67
placement in the configuration file 69
use in the configuration file 71, 89
PVM/IPCS support
COMBLOCK command 340
DEVQUE command 340
HELP command 341
IPCSDUMP command 341
IPCSSCAN command 342
LINKS command 343

PVM/IPCS support *(continued)*
MAP command 344
ROUTES command 345
SYSBLOCK command 345
TASKQUE command 346
USERQUE command 346
PVMBLD command 315
PVMCMP command 316
changing the backup language 235
compiling the message repository 238
description 316
files created by 316
PVMCONF.DAT file 68
PVMG
* Comment record 401
AUTHORIZ 403
AUTHORIZ record 403
command
AUTHORIZ 378
CDBC 380
CMD 381
DISC 382
DROP 382
INIT 383
LMSG 384
LOGMSG 385
MSG 385
QUERY 388
QUIESCE 390
RESUME 391
ROUTE 391
SELECT 393
SHOW 393
SHUTDOWN 120, 394
SPOOL 396
START 397
TRACE 398
CONFIG file 53
configuration record, APPLN 402
directory, sample 42
installation verification 55
LOCAL 406
PMSG 386
record
DUMP 403
LANG 404
LINK 404
LOGINID 407
LOGMODE 406
LU 408
MSGNOH 410
MSGOPTNS 410
OPID 411
PRVPROP 412
ROUTE 412
SELECT 413
SESSDISC 413
SESSTERM 414
use in the configuration file 90
PVMG *(continued)*
record *(continued)*
XDBC 414
PVMG LNODE record
description 405
PVMGEN command
description 317
used to enable PVM for additional languages 233
used when applying exits to PVM 226
used when changing the backup language 236
used when changing the PVM translate table 237
used when creating or updating a message repository 240
PVMGMBLD command 387
PVMPROP record, PVMG 412
PVMPROP user ID
description 251
messages sent to a 252
PVMPROP record 369
PVMPROP/CRI message protocol 252
specifying user ID for 369
PVMSNAP EXEC 257

Q

QUERY command, PVM
description 318
QUERY command, PVM 318
samples 107—109
QUERY command, PVMG 388
PVMG samples 117
QUERY command, PVMG 388
quick method of loading tape files
distribution tape 47, 48
national language feature tape 230
QUIESCE command
description 321
PRT option usage 117
QUIESCE command, PVM 321, 323
reversing effect of a PVM RESUME command 323
samples 112, 113
QUIESCE command, PVMG 390
QUIESCE command, PVMG 390
QUIT command
description 323
usage by a PVM operator 114

R

R3270 line driver
adding devices to a 72
defining clusters (CLUSTER record) 353
defining ports on a (CLPORT record) 351
description 6
R3270 line driver 71
DEFINE command (PVM) 293
LINK record 358
removing function from PVM 202

READ/BUFFER command restriction is not supported. 34
READ/MODIFIED command restriction 34
remote 3270 line driver
  adding devices to a 72
  defining clusters (CLUSTER record) 353
  defining ports on a (CLPORT record) 351
  description 6
  R3270 line driver 71
    DEFINE command (PVM) 293
    LINK record 358
  removing function from PVM 202
remote node
  routing 74
remote systems
  accessible types of 5—6
removing functions from PVM 201—202
required minidisks 45
requirements 22
  DASD 25
  for a VM/XA installation 38
reserving ports 31
RESID record, PVM
  description 370
  example 67
  use in the configuration file 71
  used with CMS communication directory 80
resource identifier, specifying 370
restricted commands 286
restrictions
  3270 IDS printers 35
  3271/3274 emulation 34
  CTCA 34
  data length 33
  for VM/ESA and VM/XA installations 37
  general 33
  notepad function 33
  port usage 31
  ROCF 36
  SYS REQUEST key 33
  TEST REQUEST key 33
RESUME command, PVM
RESUME Command, PVMG 391
RESUME record, PVM
  description 323
  samples 112, 113, 120
return codes, from MPVM command 306
reversing a QUIESCE command (RESUME command) 323
ROCF line driver
  abnormal termination handling 32
  defining 71
    DEFINE command (PVM) 293
    LINK record 358
  description of 5
  performance considerations 201
  removing function from PVM 202
  requirements 22
ROCF line driver *(continued)*
  restrictions 33, 36
  sample user session 437
  specifying type 36
  time out condition handling 78
route
  defining 324, 370
  displaying 110
  ROUTE command, PVM 324
  ROUTE record 370
  SHOW command, PVM 328
ROUTE command 324
ROUTE command, PVM
  ROUTE command, PVM 324
ROUTE command, PVMG 391
  ROUTE command, PVMG 391
ROUTE record, PVM
  description 370
  example 68
  placement in the configuration file 69
  use in the configuration file 74
  used to determine routing 74
ROUTE record, PVMG 412
  ROUTE record, PVM 412
route table, displaying 190
ROUTES (IPCS) command 345
routing 74
RUNPVM command
  description 325, 392
  samples 103—105

S

S. command prefix 255
samples
  BUFFERS record calculation 350
  CMS communications directory 80
  configuration file 66—69
  EXT record calculation 354
  I/O record calculation 356
  message repository 248
  operator commands 103—117
  PROFILE EXEC 79
  PROFILE PVM 82—83
  session control routine 217
  SET RESERVE calculation 200
saving information
  AUDIT record 77
  DUMP record 77
screen images, logging 193
screen record, in a message repository 243
security considerations
  abnormal termination 32
  eliminating exposures 30
  notepad function 29
  restricting port usage 31
  session security 30
  use of PORT record 31

SELECT command
  PVM 326
  PVMG 393
SELECT record, PVMG 413
Selection Menu
  defining PF keys for (SELECT record) 371
  setting the LOGMSG area (PVM LOGMSG command) 303
sending messages
  CP SMSG 339, 399
  LMSG 111
  LMSG (PVM) 302
  LMSG (PVMG) 118
  LOGMSG 303
  MSG 111
  MSG (PVM) 308
  MSG(PVMG) 118
  PMSG 111
  PMSG (PVM) 314
  PMSG(PVMG) 118
service staging minidisk 45
service tape 278
SESSDISC record, PVMG 413
session control exits 212—218
  command and response (DVMNUE5) 215
  intermediate node session request (DVMNUE4) 214
  origin node session request (DVMNUE3) 214
  sample 217
  session initiate data exit (DVMNUE6) 216
  session initiation (DVMNUE1) 212
  session termination (DVMNUE2) 213
session initiation exit (DVMNUE1) 212
session initiation exit (DVMNUE6) 216
session statistics exit (DVMNMES) 219, 221
session terminate function
  changing the system default 371
  security exposure 30
  setting with PASSTHRU command 311
session termination exit (DVMNUE2) 213
SESSTERM record, PVMG 414
SET command, PVM 326
SET FAVORED command 199
SET QDROP OFF command 199
SET RESERVE command 200
setting defaults
  alternate routing 370
  dump types 353
  for PASSTHRU parameters
    CMS GLOBALV command 312
    CMS SYNONYM file 312
    CP SET command 312
    NAMES file 312
  LANG record 357
  message options 365
  MSGOPTNS record 365
  national language 357
  ROUTE record 370
setting defaults *(continued)*
  routing 370
  session terminate function 371
  SESSTERM record 371
severity characters 241
SHOW command, PVM
  description 328
  sample use 110
  SHOW command, PVM 328
SHOW command, PVMG 393
  sample use 118
SHUTDOWN command
  PVM 114, 329
  PVMG 120, 394
SMSG command 339, 399
source staging minidisk
  used during feature tape installation 229
  used during PVM installation 45
special considerations for a message repository 246
specifying
  a command response language with a command prefix 255
  an alternate operator
    AUTHORIZ command, PVM 287
    AUTHORIZ record 349
  CMS Subset Mode commands (SUBSET record) 372
  default national language (LANG record) 357
  dump types (DUMP record) 353
  message requirements 78, 91
  notepad function (PASSTHRU command) 309
  session language (PASSTHRU command) 309
  session terminate string (PASSTHRU command) 309
  storage requirements 79
  target system (PASSTHRU command) 309
  temporary disconnect (PASSTHRU command) 309
specifying a pvmid 304
SPOOL command, PVMG
  description 396
standalone method of loading tape files
  distribution tape 47, 48
  national language feature tape 231—233
START command
  description 330
  LINE option usage 116
  PRT option usage 115
  PVM 330
  PVMG 397
start highlighting character 241
start variable character 241
starting
  lines 330
  printer sessions 115
  PVM 103
  remote 3270 displays and printers 115
  RUNPVM command description 325
  START command description, PVM 330

starting MPVM 304
Station Select feature requirement 23
STATUS command 332
stopping
  PVM 114
    QUIT command, PVM 323
    SHUTDOWN command,PVM 329
  PVM printer sessions 117
  remote 3270 displays and printers 117
storage considerations
  auxiliary requirements 25
  calculating BUFFERS record value 350
  calculating EXT record value 354
  calculating I/O record value 356
  default PVM size 42
storage requirements 25
substitution variables in a message repository 245
syntax diagrams 281, 376
SYSBLOCK (IPCS) command 345
system directory, for PVM 42—43
system directory, for PVMG 43—46
system extension minidisk 45, 229
system NAMES file 95
system programmer
  commands
    MPVMBLD 307
    PVMBLD 315
    PVMCMP 316
    PVMGBLD 387
    PVMGEN 317
    summary 284, 377
  tasks
    changing message destinations 236—237
    changing the PVM translate table 237
    compiling a PVM message repository 316
    developing exits 209—226
    enhancing PVM performance 199—202
    modifying PVM messages 227—249
    providing system applications 251—257
    rebuilding the MPVM user module 307
    rebuilding the PVM MODULE 315
    regenerating DVMUSI 317
    tailoring the PVM module 201—202
System Request key restriction 33
system trace, ETRACE 183

T

T. command prefix 255
tailoring PVM MODULE 201—202
tape layouts 277
  distribution tape 277
  national language feature tape 279
  service tape 278
target subtag 96
TASKQUE (IPCS) command 346
temporary disconnect function
  disconnect time limit 77
  security considerations 30
temporary disconnect function *(continued)*
  setting maximum disconnect time allowed 373
  setting your own defaults
    CMS GLOBALV command 312
    CP SET PFn command 312
terminal addresses 53
terminating
  PVM 114
    QUIT command, PVM 323
    SHUTDOWN command,PVM 329
  PVM printer sessions 117
  remote 3270 displays and printers 117
Test Request key restriction 33
time intervals
  before terminating a temporary disconnected PVM user 373
  between calls to the link statistics exit 372
  between line status checks 373
time out
  considerations 77
  handling 78
time restrictions 77
time stamp
  controlling the display of 365
  defining format of 242
  in the PVM audit file 348
  message repository records 242
  MSGOPTNS record 365
TRACE command, PVM
  TRACE command, PVM 334
TRACE command, PVMG 398
  APPLN record, PVMG 402
  TRACE command, PVMG 398
trace facility
  line driver data transfer 182
  line driver I/O 181
  PCCF 182
  PVM system activity 182
  records 183—187
  table 181
  table, ESCON 181
  the PVM multitasking supervisor 182
  TRACE command description (PVM) 334
  using a CPTRAP file 183
tracing
  line driver data transfer 182
  line driver I/O 181
  PCCF 182
  PVM system activity 182
  records 183—187
  table 181
  table, ESCON 181
  the PVM multitasking supervisor 182
  TRACE command description (PVM) 334
  using a CPTRAP file 183
tributary address requirement 23
types of trace records 183—187

U

UCOMDIR NAMES file 80
  using with APPC/VM links 80, 417
unrestricted commands 286
updating PVM
  adding links
    DEFINE command, PVM 293
    LINK record 358
    permanent 71
    temporary 109
  changing message destinations 236
  changing the backup language 235
  changing the PVM translate table 237
  creating a message repository 238—249
  developing exits 209—226
    CMS user exits 221—225
    network management exits 218—221
    session control exits 212—218
  enhancing PVM performance
    DIAG98 directory option 199
    dispatching priority 201
    ROCF considerations 201
    SET FAVORED command 199
    SET QDROP OFF command 199
    SET RESERVE command 200
    VCUNOSHR directory option 199
  installing the national language feature
    tape 228—235
  storage 29
    BUFFERS record calculation 350
    EXT record calculation 354
    I/O record calculation 356
  tailoring the PVM module 201
USER control statement 201
user NAMES file 95
user sessions
  samples
    with remote 4300 processor 437
  security considerations 31
USERQUE (IPCS) command 346
using national languages on your system 227

V

VARY command 337
VCUNOSHR directory option 199
verifying
  national language feature tape installation 233
  PVM installation 54
VM/ESA and VM/XA
  administrative considerations 37
VM/Pass-Through Facility
  See PVM
VMDUMP type dump
  specifying on DUMP record 354
  using to create a problem report 189
VMFASM, using to assemble PVM 225
VMSERV exec 143
VMSES/E
  servicing PVM 141
VTAMLST file 53

X

XA virtual machine mode 37
XDBC record, PVMG 414

Readers' Comments

VM/Pass-Through Facility
Administration and Operation
Version 2

Publication No. SC24-5557-01

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.

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.

No postage stamp necessary if mailed in the U.S.A. (Elsewhere, an IBM office or representative will be happy to forward your comments or you may mail directly to the address in the Edition Notice on the back of the front cover or title page.)

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

Name _____ Address _____

Company or Organization _____ _____

Phone No. _____ _____

IBM®

Fold and Tape **Please do not staple** Fold and Tape

NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS MAIL PERMIT NO. 40 ARMONK, NEW YORK

POSTAGE WILL BE PAID BY ADDRESSEE

International Business Machines Corporation
Information Development
Department G60
1701 North Street
ENDICOTT NY 13760-5553

Fold and Tape **Please do not staple** Fold and Tape

File Number: S370/4300-39
Program Number: 5684-100

SC24-5557-01