

File No. S370-34
Order No. GC19-6212-0

Program Product

**IBM Virtual
Machine/System Product:
Operating Systems in
a Virtual Machine**

MANUAL INFORMATION...
SERIAL...
DATE...
PAGE...
REVISION...
PUBLISHED...

File No. S370-34
Order No. GC19-6212-0

Program Product

IBM Virtual Machine/System Product: Operating Systems in a Virtual Machine

Program Number 5664-167

This publication is for system programmers who plan to use System/360 or System/370 operating systems under the control of VM/SP. It is also for VM/SP system programmers who plan to use these operating systems or VM/SP under the control of VM/SP. The publication describes how to use VM/SP functions more efficiently when running operating systems under VM/SP. The book also provides system planning and operating considerations when running VM/SP, DOS/VS, and OS/VS under VM/SP.

Many thoughts expressed in this publication are contributions from VM/370 customers and IBM field personnel. As such, these thoughts may not have been formally tested by IBM, and therefore, may prove more useful for some installations than others.

PREREQUISITE PUBLICATION

*IBM Virtual Machine/System Product
Introduction, Order No. GC19-6200*

The IBM logo, consisting of the letters 'IBM' in a bold, sans-serif font, with each letter formed by a series of horizontal bars of varying lengths.

Notice: The term VM/SP, as used in this publication, refers to VM/SP when used in conjunction with VM/370 Release 6.

The recommendations in the VM/SP under VM/SP area and in the DOS/VS and OS/VS areas of this publication are meant to help an installation generate operating systems that run more efficiently under VM/SP.

VM/370 users made many of these recommendations, and their suggestions have not been submitted to any formal IBM test. As a result, potential users should evaluate the applicability of the recommendations to their installations before implementation.

First Edition (September 1980)

This edition (GC19-6212-0) applies to the IBM Virtual Machine/System Product, and to all subsequent releases unless otherwise indicated in new editions or Technical Newsletters. Changes are continually made to the information contained herein; before using this publication in connection with the operation of IBM systems, consult the IBM System/370 and 4300 Processors Bibliography, GC20-0001, for the editions that are applicable and current.

It is possible that this material may contain reference to, or information about, IBM products (machines and programs), programming, or services that are not announced in your country. Such references or information must not be construed to mean that IBM intends to announce such IBM products, programming, or services in your country.

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

A form for readers' comments is provided at the back of this publication; if the form has been removed, comments may be addressed to IBM Programming Publications, Dept. G60, P.O. Box 6, Endicott, New York, U.S.A. 13760. IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation whatever. You may, of course, continue to use the information you supply.

Preface

This publication is for system programmers who run OS, DOS, DOS/VS, DOS/VSE, OS/VS1, OS/VS2 SVS, OS/VS2 MVS, VM/370, and VM/SP under the VM/SP control program. It is also for VM/SP system programmers who plan to use these System/360 or System/370 operating systems under the control of VM/SP.

Users of the control program (CP) should refer to the IBM Virtual Machine/System Product CP Command Reference for General Users, Order No. SC19-6211, and IBM Virtual Machine/System Product Operator's Guide, Order No. SC19-6202.

Users of the conversational monitor system (CMS) should refer to the IBM Virtual Machine/System Product CMS User's Guide, Order No. SC19-6210.

Users of the remote spooling communications subsystem (RSCS) should refer to the IBM Virtual Machine Facility/370: Remote Spooling Communications Subsystem (RSCS) User's Guide, Order No. GC20-1816.

Users of the interactive problem control system (IPCS) should refer to the IBM Virtual Machine Facility/370: Interactive Problem Control System (IPCS) User's Guide, Order No. GC20-1823.

ORGANIZATION

This publication contains these sections:

- "Section 1. General Considerations" provides both introductory and general usage information. The introductory information briefly describes the features of VM/SP as they apply both to the virtual machine and to the operating system that is running in it. The general usage information discusses those aspects of running operating systems under VM/SP that are common to all systems. It also describes how to use VM/SP functions more efficiently when running operating systems under VM/SP.
- "Section 2. VM/SP in a Virtual Machine" explains how to test and update a VM/SP system that is itself operating under VM/SP.
- "Section 3. DOS/VS in a Virtual Machine" provides operating information specific to running DOS, DOS/VS, and DOS/VSE in a virtual machine. It contains system planning considerations for generating and using these systems under VM/SP, rather than stand-alone.
- "Section 4. OS/VS in a Virtual Machine" provides operating information specific to running OS, OS/VS1, and OS/VS2 in a virtual machine. It contains system planning considerations for generating and using these systems under VM/SP, rather than stand-alone.

TERMINOLOGY

In this publication, the following terminology is used:

- "2305" refers to the IBM 2305 Fixed Head Storage, Models 1 and 2.
- "3270" refers to a series of display devices, namely, the IBM 3275, 3276, 3277, 3278, and 3279 Display Stations. A specific device type is used only when a distinction is required between device types.
- "3330" refers to the IBM 3330 Disk Storage, Models 1, 2, and 11; the IBM 3333 Disk Storage and Control, Models 1 and 11; and the 3350 Direct Access Storage operating in 3330/3333 Model 1 or 3330/3333 Model 2 compatibility mode.
- "3340" refers to the IBM 3340 Disk Storage; Models A2, B1 and B2; and, the 3344 Direct Access Storage, Model B2.
- "3350" refers to the IBM 3350 Direct Access Storage, Models A2 and B2, in native mode.
- "FB-512" refers to the IBM 3310 and 3370 Direct Access Storage Devices. These devices are supported by VM/SP.
- "3380" refers to the IBM 3380 Direct Access Storage device.
- "3800" refers to the IBM 3800 Printing Subsystem.
- "Cylinder" describes space on Direct Access Storage Devices (count-key-data devices) supported by the VM/SP system control program.

- "Block" describes DASD space on FB-512 devices supported by VM/SP.
- The term "area" may appear in text when there is no need to differentiate between DASD space on count-key-data devices or FB-512 devices.
- Unless otherwise noted, the term "VSE" refers to the combination of the DOS/VSE system control program and the VSE/Advanced Functions program product.

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 may be referred to as DOS disks.

The DOS like simulation environment provided under the CMS component of the VM/System Product, continues to be referred to as CMS/DOS.

- "Records" describes a spool file generated to represent physical card decks.

Any information about display terminal usage also applies to the IBM 3138, 3148, and 3158 Display Consoles when used in display mode, unless otherwise noted.

Any information pertaining to the IBM 3284 or 3286 printer also pertains to the IBM 3287, 3288, and 3289 printers, unless otherwise noted.

Any information pertaining to the IBM 2741 terminal also applies to the IBM 3767 terminal, Model 1, operating as a 2741, unless otherwise specified.

Information on the IBM 3380 Direct Access Storage device is for planning purposes only, until the availability of the product.

For a glossary of VM/SP terms, refer to IBM Virtual Machine/System Product Glossary and Master Index, Order No. GC19-6207.

PREREQUISITE PUBLICATIONS

IBM Virtual Machine/System Product Introduction, Order No. GC19-6200

COREQUISITE PUBLICATIONS

The reader must also have a basic knowledge of the operating systems he will be using under VM/SP. For the titles and abstracts of the appropriate publications, refer to the latest IBM System/370 and 4300 Processors Bibliography, Order No. GC20-0001.

IBM Virtual Machine/System Product

System Programmer's Guide, Order No. SC19-6203

Planning and System Generation Guide, Order No. SC19-6201

CP Command Reference for General Users, Order No. SC19-6211

CMS User's Guide, Order No. SC19-6210

CMS Command and Macro Reference, Order No. SC19-6209

Operator's Guide, Order No. SC19-6202

Terminal User's Guide, Order No. GC19-6206

ASSOCIATED PUBLICATIONS

IBM Virtual Machine/System Product OLTSEP and Error Recording Guide, Order No. SC19-6205

OS/VS, DOS/VSE, VM/370 Environmental Recording Editing and Printing (EREP) Program, Order No. GC28-0772

SUBMITTING INFORMATION ON RUNNING OPERATING SYSTEMS IN A VIRTUAL MACHINE

You are welcome to submit recommendations and hints about generating and running operating systems in a virtual machine for possible inclusion in this publication. You can write a letter, or use either the Suggestion Input Form or the Reader's Comment Form at the back of this publication. Send your recommendations to:

IBM Corporation
 VM/SP Publications Department
 Dept. G60, P.O. Box 6,
 Endicott, New York 13760

It is understood that IBM and its affiliated companies shall have the nonexclusive right, at their discretion, to use, copy, and distribute any or all submitted information or material, in any form, for any and all purposes, without any obligation or commitment to the submitter, and that the submitter has the right to submit such information or material upon such a basis.

When submitting recommendations, please indicate the type of operating system (that is, DOS/VS, VS1) and the release level that is used under VM/SP.

GUIDELINES AND PROCEDURES

Suggestions for generating and running operating systems can cover any topic currently in the book or can address areas that you think should be added to this publication. Submit input in whatever manner and format is most convenient, but ensure that it is legible, understandable, and follows these guidelines:

- Do not include changes to object modules since such modules are likely to change over time.
- Do not describe a problem unless you have an appropriate solution, circumvention, or alternative included. Tips on things you have to watch out for, or unusual circumstances that can occur in virtual machine operation are, however, suitable topics.
- If you describe useful EXEC procedures for CMS or other programs, test them out to ensure that they work.

Note: The recommendations in the DOS/VS and OS/VS areas of this publication are meant to help an installation in generating operating systems to run more efficiently under VM/SP and also contain operational considerations or hints when using virtual machines. Many of these recommendations were suggested by VM/370 users and have not been submitted to any formal IBM test. As a result, potential users should evaluate the applicability of the recommendations to their installation before implementation.

Contents

SECTION 1. GENERAL CONSIDERATIONS.	1	2305 Devices	61
Virtual Machine Resources.	2	Accessing a Virtual VM/SP System	61
Virtual Machine Operating Systems.	2	CP Disks for the Virtual Machine	61
Other Programs and Systems	4	Virtual IPL and Operation.	63
Error Recording and Analysis	4	Accessing Devices.	63
Unsupported Devices.	4	Spooling Considerations.	64
Programming Considerations	5	Example -- Running VM/SP Under VM/SP	65
Paging Factors	5	Summary.	76
Reducing Paging Activity	5	SECTION 3. DOS/V S IN A VIRTUAL MACHINE	77
Abnormal Terminations in a Virtual Machine	6	System Generation Recommendations.	78
Reducing a Virtual Machine's I/O Operations.	6	VM/SP Recommendations.	78
Virtual Machine Options.	6	DOS/V S Recommendations	79
Data Transfer Using VMCF	8	DOS/V S Accounting.	81
Data Transfer Using IUCV	8	Generating DOS/V S Under VM/SP.	82
BTAM Autopoll Channel Programs	8	Building System Generation Job Streams	83
Special Considerations for Multiprogramming Systems Under VM/SP.	9	Copying Distribution System to Disk.	84
VM/V S Handshaking.	9	Readvancing the Interim System.	85
The Diagnose Interface	10	Assembling and Loading a New Supervisor.	85
Page Waits	10	Adding I/C Modules to DOS/V S (DOS/V S Release 34 and earlier only).	88
I/O Waits.	10	Final Housekeeping	89
Spooling	11	Sample DOS/V S Directory Entries.	89
Processor Model and Channel Model Dependencies.	12	Accessing DOS/V S	90
Specifying Virtual Machine Consoles.	14	Sharing the DOS/V S System Residence Volume.	90
Virtual Machine I/O Management	15	Standard Label Cylinder.	91
VM/SP Alternate Path Support	17	Using Virtual Unit Record Devices.	92
Operating Systems Using DASD Reserve/Release	18	Defining the Operator's Console.	92
Shared DASD.	18	Preparing Jobs for a DOS/V S Virtual Machine	93
Full Screen Console Support for Virtual Operating Systems	21	Loading DOS/V S	94
Alternating Between Operating Systems.	23	IPL from the Console	94
Multiple-Access Virtual Machines	25	IPL from the Card Reader	97
The ASP Virtual Machine.	30	DOS/V S Operation	98
Shadow Table Maintenance Support	32	Starting a Job Stream.	99
Multiple Shadow Table Support.	32	When a Job Is Finished	99
Selective Invalidation	32	Communicating with CP.	99
Shadow Table Bypass for V=R Users.	33	Running Batch DOS/V S Under VM/SP	101
Shadow Table Bypass for V=V Users.	34	Alternating Between CMS and DOS/V S Under VM/SP	102
Performance Guidelines	38	Loading CMS into a Virtual Machine	102
Performance Measurements	42	Using the CMS Editor To Prepare Jobs	102
Emphasizing Interactive Response Times	42	Issuing SPOOL Commands To Control Unit Record Devices	103
Generation Procedures Under VM/SP.	43	Punching CMS Files	104
Creating VM/SP Directory Entries	43	Initializing DOS/V S.	104
Unique Directory Entry Considerations.	46	Signaling DOS/V S To Read the Job Stream.	105
Defining Virtual Devices	53	Reloading CMS into a Virtual Machine	106
AUTOLOG Facility	54	Examining DOS/V S Virtual Machine Output.	106
Sample Directory Entries	56	Using EXEC Procedures.	107
Summary.	57	Using More Than One Virtual Machine.	108
SECTION 2. VM/SP IN A VIRTUAL MACHINE.	59	Accessing CMS.	109
VM/SP Directory Definition	59	Disconnecting CMS.	109
Virtual Machine Configuration.	60	Logging Onto DOS/V S.	109
Defining a Console for VM/SP in a Virtual Machine	60	Returning to CMS	110
CMS System	61		

Disconnection Considerations	111	Machine	137
Developing and Testing Programs to Run in a DCS/VS Virtual Machine	112	Job Entry and Output Retrieval	138
Summary	114	Loading OS/VS	138
SECTION 4. OS/VS IN A VIRTUAL MACHINE. .115		OS/VS Operation	139
System Generation Recommendations	116	Communicating with CP	140
VM/SP Recommendations	116	Using OS/VS in Batch Mode Under VM/SP .141	
OS/VS Recommendations	117	Alternating Between CMS and OS/VS Under VM/SP	141
OS/VS1 Recommendations	118	Loading CMS into a Virtual Machine . .142	
VM/VS Handshaking for VS1	119	Using the CMS Editor To Prepare Job Streams	142
IBM 3850 Mass Storage System Considerations	124	Issuing Spool Commands to Control Unit Record Devices	142
Generating OS/VS Under VM/SP	124	Punching CMS Files	143
Preparing for OS/VS System Generation.125		Initializing OS/VS	143
Initializing the Starter System Volume	126	Reloading CMS into a Virtual Machine .143	
Restoring the Starter System to Disk .128		Examining OS/VS Virtual Machine Output	144
Loading the Starter System	128	Dynamic SCP Transition to or from Native Mode	144
Restoring Distribution Libraries to Disk	128	Operating Procedures	145
Creating Card Decks for Utility Programs	129	Error Recording	146
Initializing the System Residence Volume	129	Using More Than One Virtual Machine . .146	
VM/SP Utility Programs	130	Disconnection Considerations	147
Stage I Processing in the CMS Virtual Machine	130	Developing and Testing Programs to Run in an OS/VS Virtual Machine	149
Stage II Processing	132	OS/VS2 MVS Uniprocessor Under VM/SP . .149	
Final Housekeeping	133	Use of Single Processor Mode in AP and MP Systems	149
Sample OS/VS Directory Entries	133	Operating Procedures	150
Accessing OS/VS	135	Error Recording	151
Using Virtual Devices	136	Taking a VM/SP Dump	151
Defining the Operator's Console136		Summary	152
Using the VM/SP Spool File System . . .137		INDEX	153
Preparing Jobs for an OS/VS Virtual			

FIGURES

Figure 1.	Summary of VM/SP Reserve/Release Support.....	21	Figure 9.	A Virtual 2703 TCU Controlling Remote 3270 Terminals.....	29
Figure 2.	OS Job Stream Transfer.....	24	Figure 10.	Two ASP Virtual Machines.....	30
Figure 3.	Directory Entry for Alternating Between Operating Systems.....	24	Figure 11.	One Real and One Virtual ASP Machine.....	31
Figure 4.	Virtual Devices: Local 3270 Terminals.....	26	Figure 12.	Relationship of VM/SP Directory Entries.....	45
Figure 5.	Virtual Devices: Remote Terminals.....	26	Figure 13.	Virtual=Real Machine.....	49
Figure 6.	A Virtual VM/SP Multiple-Access Systems.....	27	Figure 14.	Virtual Machine for DOS/VS System Generation.....	82
Figure 7.	VM/SP Directory Entry for a Multiple-Access Virtual Machine.....	27	Figure 15.	IPL DCS/VS and Execute a Job in the Card Reader.....	96
Figure 8.	A Communications Test System.....	28	Figure 16.	Virtual Machines of OS/VS System Generation.....	125
			Figure 17.	Sample IPL of VS1 under VM/SP.....	139

Section 1. General Considerations

The Virtual Machine/System Product (VM/SP) provides an easy, convenient way to use a single terminal to run other operating systems, such as DOS/VS, OS/VS1, or OS/VS2. With VM/SP, system programmers can test a new application program with an operating system, or they can develop and test new operating system releases. They can do this work on any shift, and they can do it isolated from any work that is running concurrently elsewhere in the system. This isolation is done by the use of a virtual machine.

A virtual machine is a functional equivalent of an IBM System/370 computing system. Each virtual machine has the functional equivalent of a real processor, main and auxiliary storage, and I/O devices. Because VM/SP only simulates these functions, this simulated machine is referred to as a "virtual" machine. VM/SP manages the functions of a real IBM System/370 in such a way that virtual machines are available to multiple concurrent users.

Before running any operating system in a virtual machine, an installation should consider:

- How can application programs operate efficiently in a virtual machine?
- How it can reduce a virtual machine's I/O operations?
- Which services are available for performance and communication for both VM/SP and a virtual machine?
- What special considerations are there for multiprogramming operating systems under VM/SP, such as DOS/VS or OS/VS?
- What operating system functions and devices does VM/SP support and not support?
- How to generate an operating system under VM/SP?
- How can virtual machines access the VM/SP system?

In answering these questions, this publication assumes that readers have a basic understanding of VM/SP concepts and functions as described in the VM/SP Introduction. It also assumes that they have a basic understanding of whatever operating system they are running under VM/SP.

What virtual machine resources and operating systems can be used under VM/SP?

VIRTUAL MACHINE RESOURCES

Virtual machine resources are either shared among users or allocated to users alternately for a specified time period. The resources to be allocated to any particular virtual machine are specified in the VM/SP directory entries for that virtual machine. The directory entries for all virtual machines make up the VM/SP directory file that is usually located on the VM/SP system residence volume. When a user obtains access to the VM/SP system, a virtual machine is created based upon that user's directory entry. The user can then load any of the supported operating systems and begin processing.

For a virtual machine to begin processing under VM/SP, what functions does VM/SP itself provide?

VM/SP Components

The VM/SP system consists of:

- The control program (CP): CP controls the resources of a computer such that multiple virtual machines or computing systems appear to exist.
- The conversational monitor system (CMS): CMS provides a wide range of conversational and time-sharing facilities. By using CMS, an installation can create and manage files, and compile, test, and execute problem programs.
- The remote spooling communications subsystem (RSCS): RSCS transfers spool files between VM/SP users and remote locations over telecommunication lines.
- The interactive problem control system (IPCS): IPCS provides VM/SP problem analysis and management facilities, including problem report creation, problem tracking, and CP abend dump analysis.

For an overview of these VM/SP concepts, refer to VM/SP Introduction.

Note: Throughout this publication, the term 'VM/SP' refers to the VM/SP program package when you use it in conjunction with VM/370 Release 6. The terms 'CP' and 'CMS' refer to the VM/370 components enhanced by the functions included in the VM/SP package. Any referral to 'RSCS' and 'IPCS' unless otherwise noted, is related to the VM/370 components unchanged by the VM/SP package.

VIRTUAL MACHINE OPERATING SYSTEMS

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

The operating systems that can execute in virtual machines are:

Batch or	Single-User	Interactive	Multiple-Access
	DOS		VM/SP
	DOS/VS		VM/370
	DOS/VSE		Time Sharing
	OS/PCP		Option of OS
	OS/MFT		DOS/VSE with
	OS/MVT		VSE/ICCF (5746-
	OS/VS1		TS1)
	OS/VS2 SVS		
	OS/VS2 MVS		<u>Conversational</u>
	OS-ASP		CMS
	RSCS		

CP provides each of these with virtual device support and virtual storage. The operating systems themselves execute as though they are controlling real devices and real storage, but they must not violate any of the restrictions listed in VM/SP Planning and System Generation Guide.

Batch and Single-User Interactive Systems

With the exception of OS/PCP, all the batch or single-user systems are multiprogramming systems. However, when operating in a virtual machine under VM/SP, the user has the choice of running either multiple partitions in one virtual machine similar to stand-alone operation or single partitions in multiple virtual machines. When running multiple partitions in one virtual machine, multiprogramming and unit record spooling is done by both the operating system and VM/SP. This may decrease the overall efficiency of the virtual machine. When running single partitions in multiple virtual machines, the need for multiple virtual storage spaces places a burden on auxiliary storage. However, using shared systems reduces this burden.

Multiple-Access Systems

Each multiple-access system operates in one virtual machine and supports multiple interactive users. The multiple-access virtual machine must first gain access to VM/SP (by using the LOGON command). Subsequently, interactive users can connect to the multiple-access system (either by using the DIAL command or by using a terminal on a dedicated line). Communication between the two is carried out by using the command language of the multiple-access system.

Conversational Monitor System

The conversational monitor system (CMS) is a VM/SP component that provides a wide range of conversational and time-sharing facilities. Together with the control program of VM/SP, it provides a time-sharing system suitable for direct problem solving and program development. By using CMS, a virtual machine user can create, update, and manipulate files. The user can also compile, test, and execute problem programs.

The CMS interactive capabilities are extended to DOS/VS and VSE environment users by using either the CMS/DOS environment or CMS. For OS/VS users, a combination of CMS commands and CMS simulation of OS macro instructions provides similar interactive capabilities.

For information on using the CMS virtual machine, refer to VM/SP CMS User's Guide and VM/SP CMS Command and Macro Reference.

OTHER PROGRAMS AND SYSTEMS

For information about other programs and systems that have been used under VM/SP, request information on Installed User Programs (IUPs), Program Products (PPs), and Field Developed Programs (FDPs) from your local IBM branch office. For a list of these programs, refer to VM/SP Planning and System Generation Guide.

ERROR RECORDING AND ANALYSIS

The operating systems commonly run in virtual machines all use SVC 76 to write error records to the error recording data sets. However, in a virtual machine VM/SP intercepts SVC 76 and records the error in its own error recording area. Therefore, error records from all operating systems reside in this one centralized error recording area. To access the recorded data, use the CMS CPEREP command. For further information about error recording, formatting output from the error recording area with Service Record File devices, and CPEREP refer to VM/SP QLTSEP and Error Recording Guide.

UNSUPPORTED DEVICES

Virtual machine users may be able to use I/O devices that VM/SP does not support. An unsupported device is a device type that is not listed in the DEVTYPE operand of the RDEVICE macro instruction. To use an unsupported device, a user must attach or dedicate the device to a virtual machine. A dedicated device is one that is not shared among users, but is used exclusively by one user. However, VM/SP supports these dedicated devices only under these conditions:

- No timing dependencies exist in the device or the program.
- No dynamically modified channel programs exist in the access method, except when OS ISAM or OS/VS TCAM Level 5 are used.
- No special functions need to be provided by VM/SP.
- None of the other CP restrictions are violated. (Refer to the VM/SP restrictions list in the VM/SP Planning and System Generation Guide.)
- The device is generated into the VM/SP nucleus (by using the RDEVICE macro instruction with the appropriate CLASS operand).

I/O devices that are part of a virtual machine's configuration require real device equivalents. However, exceptions to this rule are:

- Unit record devices, which VM/SP can simulate by using spooling techniques.
- Virtual 2311 disks which VM/SP can map onto 2314 or 2319 disks. One to two full 2311 units can be mapped onto a 2314 or 2319 disk in this manner.

Programming Considerations

New application programs should be designed to operate efficiently in a paging environment. Whenever possible, use VM/SP paging instead of DOS/VS or OS/VS paging. That is, make the DOS/VS partitions and OS/VS regions virtual=real (V=R) and large enough to contain the largest jobs. Eliminate all overlays and, if possible, combine into one larger job any multistep jobs that use temporary DASD storage.

PAGING FACTORS

Installations should be aware that the following factors affect the performance of a virtual machine:

- The frequency of real interruptions that occur
- The frequency and type of privileged instructions executed
- Whether the virtual machine assist or VM/370 extended control-program support hardware is on the machine and enabled by both the system operator and by the user
- The frequency of START I/O (SIO) instructions
- Locality of reference for paging activity within virtual storage
- The amount of fixed head paging space
- The location of the paging areas on DASD
- Whether the enhanced page migration scheme polls preferred paging areas using moveable heads

These factors are in addition to those described under the topic "Performance Guidelines" in this section.

REDUCING PAGING ACTIVITY

When a virtual machine refers to virtual storage addresses that are not in real storage, a page fault (and paging activity) occurs. Routines that have widely scattered storage references tend to increase the paging load caused by this virtual machine.

When possible, modules dependent upon each other, as well as the related reference tables, constants, and literals, should be located in the same 4K page. Infrequently used routines, such as those that handle

unusual error conditions, should not be placed near main routines. To minimize paging, reentrant coding techniques should be used whenever possible.

ABNORMAL TERMINATIONS IN A VIRTUAL MACHINE

In VM/SP, there are three levels of storage:

- First level storage - real storage
- Second level storage - virtual storage that VM/SP creates and manages for a virtual machine
- Third level storage - virtual storage that a virtual storage operating system, such as MVS, creates and manages when running in a virtual machine

Whenever possible with a virtual storage operating system, use its dumping procedure instead of VM/SP's. The CP dump program does not print out third level storage pages (that is, V=V regions or partitions of OS/VS and DOS/VS machines) in the correct sequence. Pages that happen to be stored on the OS/VS or DOS/VS paging disk are not printed at all. Refer to VM/SP CP Command Reference for General Users for information about CP dump commands. Also, several special formatting dump programs are available to help a user trace through DOS/VS and OS/VS control blocks. For more debugging information, refer to the VM/SP System Programmer's Guide.

REDUCING A VIRTUAL MACHINE'S I/O OPERATIONS

The number of SIO instructions executed by a virtual machine may be substantially reduced by:

- Using 4K byte blocking factors for I/O areas
- Preallocating the DASD space for OS or OS/VS work data sets
- Using virtual storage instead of DASD work files for smaller temporary files
- Building temporary files in virtual storage and letting VM/SP page out the data (if needed)
- Omitting virtual printers, punches, and readers from each partition or region in a virtual machine because records for these devices are unblocked
- Using the virtual operating system's spooling subsystem (such as POWER/VS or JES) because these spooling subsystems use large I/O areas and long chains of CCWs

VIRTUAL MACHINE OPTIONS

VM/SP provides several optional services to virtual machines. Specify these options either in the OPTION control statement of the VM/SP directory program or, for many options, in the CP SET command.

The BMX Option

The BMX (virtual block multiplexer) option allows an operating system running in a virtual machine to overlap multiple SIO requests on a specified channel path. The selector channel mode is the normal (and default) channel mode for virtual machines. When the BMX option is given control, it applies to all channels in the virtual machine, except to channel 0 and channels that have a channel-to-channel adapter (CTCA). This option can be specified regardless of whether block multiplexer channels are attached to the processor. The CP DEFINE command can redefine the channel mode for a virtual machine.

The ECMODE Option

The ECMODE option allows the virtual machine to use the complete set of virtual System/370 control registers and the dynamic address translation feature of the System/370. Programming simulation and hardware features are combined to allow use of all the available features in the hardware. This option is required both for DOS/VS, OS/VS1, OS/VS2, and VM/370 virtual machines and when executing VM/SP under VM/SP. It is also required when executing the generalized trace facility (GTF) under OS/MVT.

The ISAM Option

The ISAM option allows the virtual machine to execute the self-modifying CCW command sequences generated by the OS ISAM modules in OS PCP, MFT, or MVT. This option is not required for the proper functioning of ISAM in DOS or OS/VS. However, the ISAM option is required under one of these two conditions: (1) if ISAM is run in the virtual=real area of an OS/VS virtual machine, or (2) if VM/VS handshaking is active. This option does not permit other types of self-modifying CCW sequences to function.

Certain ISAM channel programs that execute under OS/PCP, MFT, MVT, or in a V=R region of OS/VS use a self-modifying operation that is not allowed under normal VM/SP processing. With the ISAM option selected, VM/SP can scan the specific ISAM channel program to handle the self-modifying sequence properly.

Only those users with the ISAM option in their VM/SP directory entry or who have issued the CP SET ISAM ON command have their CCW strings checked for self-modifying operation; thus, not all users incur the additional VM/SP overhead. This option is not needed for DOS and OS/VS ISAM when run in a V=V region.

The REALTIMER Option

The REALTIMER option updates a virtual machine interval timer when that virtual machine is in a self-imposed wait state. This option is required for virtual machines running systems that wait for a timer interruption to continue processing.

Specifying the Options

The REALTIMER, ISAM, and ECMODE options increase the amount of VM/SP overhead incurred by the virtual machines using them. Therefore, do not specify them for a virtual machine unless they are required. These options can be specified either in the OPTION statement in the VM/SP directory or by using the CP SET command. If a particular situation requires an option only occasionally, use the CP SET command and not the OPTION statement. In this way, the additional overhead is incurred only while the option is in effect.

For more information about specifying these and the other options in the OPTION control statement, refer to the topic "Creating VM/SP Directory Entries" in this section.

DATA TRANSFER USING VMCF

Virtual machines can communicate and exchange data with other virtual machines by using the virtual machine communication facility (VMCF). VMCF is the interface among communicating virtual machines. To initiate a VMCF function, the operating system in the virtual machine must issue a DIAGNOSE instruction. For a detailed description of the DIAGNOSE instruction, refer to VM/SP System Programmer's Guide.

DATA TRANSFER USING IUCV

Virtual machines can communicate and exchange data with other virtual machines by using the inter-user communications vehicle (IUCV). This is accomplished through the use of the IUCV macro instruction. IUCV allows messages to be presented to the virtual machine by polling functions or via external interrupts. For a detailed description of both IUCV functions and the IUCV macro instruction, refer to VM/SP System Programmer's Guide.

BTAM AUTOPOLL CHANNEL PROGRAMS

If an operating system is executing BTAM channel programs, VM/SP checks each BTAM autopoll CCW string to see if it has been dynamically changed. It does this each time the string is executed. To bypass this checking, issue the CP command:

```
set autopoll on
```

Whenever the BTAM autopoll CCWs are modified, OS/VS1 Release 6 and DOS/VS Release 34 with the Advanced Functions-DOS/VS Program Product (Program No. 5746-XE2) use the DIAGNOSE instruction code X'28' to notify VM/SP.

The combination of the SET AUTOPOLL ON command and the use of the diagnose interface reduces VM/SP overhead and improves the overall performance for that particular user. However, both of these facilities must be active.

If a user has specified SET AUTOPOLL ON and the operating system does not use the diagnose interface, a channel program modification goes undetected. The results are unpredictable.

If a user has specified SET AUTOPOLL OFF and the operating system uses the diagnose interface, the unnecessary checking results in performance degradation.

Note: Set DOS/VS AUTOPOLL on when using VM/VS handshaking.

Special Considerations for Multiprogramming Systems Under VM/SP

When a multiprogramming operating system such as OS/VS or DOS/VS is run in a virtual machine, its resource-management algorithms interact with those of VM/SP -- especially when the virtual operating system has a page wait or an I/O wait. Multiprogramming operating systems use these two methods to interact with VM/SP:

- VM/VS handshaking
- The diagnose interface

This topic discusses these methods and also those aspects of running an operating system under VM/SP that apply only in certain unique situations.

VM/VS HANDSHAKING

VM/VS handshaking permits instructions issued by an operating system in a virtual machine to be processed directly by the processor. It also permits VM/SP to simulate privileged instructions. VM/VS handshaking is available for these operating systems running in virtual machines under VM/SP:

- DOS/VS Release 34 with the Advanced Functions-DCS/VS Program Product (5746-XE2)
- DOS/VSE with the VSE/Advanced Functions Program Product (5746-XE8)
- VS1 Release 4 and subsequent releases

Handshaking for DOS/VS

DOS/VS Release 34 with the Advanced Functions-DOS/VS Program Product (5746-XE2) uses handshaking. For further details, refer to the appropriate DOS/VS program product publications. DOS/VSE with the VSE/Advanced Functions Program Product (5746-XE8) uses VM/VS handshaking (also known as the DOS/VSE-VM/370 linkage facility). For further details, refer to VSE/Advanced Functions General Information, GC33-6106.

Handshaking for VS1

Although handshaking is a system generation feature for VS1, it is active only when VS1 is under the control of VM/SP. It is disabled when that same VS1 operating system is run on a real machine. For details about VM/VS handshaking for VS1, refer to the "OS/VS in a Virtual Machine" section in this publication.

THE DIAGNOSE INTERFACE

The diagnose interface, by use of the DIAGNOSE instruction, permits operating systems running in virtual machines under VM/SP to communicate easily and efficiently with VM/SP.

By inserting DIAGNOSE instructions where appropriate in the operating system's code, several functions can be requested by a virtual machine. Details about how to use the DIAGNOSE instruction to request these functions are in the VM/SP System Programmer's Guide.

PAGE WAITS

If, during its execution, an OS- or DOS-created task or program must wait for a VM/SP service such as a virtual storage page, VM/SP marks the virtual machine nondispatchable even though other partitions or tasks in that virtual machine may be ready and available for processing. Those other tasks in the virtual machine cannot be dispatched by the operating system until the VM/SP page wait is satisfied. Thus, the highest priority program of the virtual machine gets almost all of the processor time allocated to that virtual machine, if it can use the time. Therefore, programs running in the other partitions experience significant degradation.

When multiprogramming systems must be executed in a virtual machine, make the partitions or regions as large as practical and execute all jobs V=R. Also, consider using the VM/SP virtual=real option, reserved page frames, or locked pages. When using the VM/SP virtual=real option, it eliminates paging for one virtual machine, but this may adversely affect the paging performance of other virtual machines. The reserved page frames option tends to keep the most active pages in storage, and the locked pages option locks the specified pages in storage.

Note: If the region size is made too large, certain programs, such as the OS sort/merge program, do not run efficiently.

I/O WAITS

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

To set the priority of a virtual machine, the VM/SP system operator can issue the CP command SET PRIORITY. A low priority value gives the virtual machine a higher priority, and this priority ensures that VM/SP dispatches the virtual machine for execution more frequently than other virtual machines.

To ensure that the lower priority DOS or OS tasks have a chance to execute, installations can use the favored execution option. This option reduces the effect of a variable system load on the favored virtual machine. It allows an installation to modify the normal VM/SP scheduling algorithms and forces VM/SP to devote more of its processor time to a given virtual machine. The option causes VM/SP to keep the specified virtual machine in the active queue, unless it becomes nonexecutable. To obtain this option for a specific virtual machine, the system operator must issue the CP command SET FAVORED.

To guarantee that a certain amount of processor time is made available to a virtual machine, installations can use the favored execution option with the percentage value specified in the SET FAVORED command.

For more details about these performance options, refer to the VM/SP System Programmer's Guide.

SPOOLING

Most multiprogramming operating systems, such as DOS/VS and OS/VS, have their own spooling subsystems (such as POWER, POWER/VS, HASP, or JES). Because VM/SP also provides its own spooling, double spooling can occur. Thus, should an installation:

- Use only the operating system's spooling subsystem?
- Use only VM/SP's spooling?
- Use double spooling?

If an installation has a significant amount of printing or punching to do, it may appear that one of the spooling subsystems should be eliminated. This is not necessarily true. In fact, if the multiprogramming operating system's spooling subsystem blocks its output (as does VS1), the most efficient spooling arrangement is usually to let both VM/SP and VS1 spool.

Note: Virtual operating systems cannot take advantage of the RAS enhancements contained in 3800 hardware engineering change level 454846. VM/SP ignores this engineering change level at real print time on a 3800.

Spooling Recommendations

If DASD space is not a limiting factor, use double spooling. If possible, generate a stripped-down version of the virtual machine's spooling subsystem, eliminating those functions not used by that virtual machine. Make the I/O buffer sizes as large as possible to cut down on SIO instructions.

If an installation has only enough DASD spooling space for one spooling subsystem and if only one virtual machine generates significant amounts of spooled output, then let that virtual machine do the spooling. However, if many virtual machines spool data and must use a common pool of unit record devices, then an installation should probably let VM/SP do the spooling.

Closing Spool Files

Output spool files are not scheduled for the real printer or punch devices until one of these actions occur:

- The user logs off, or VM/SP forces the user off.
- The user loads an operating system via the CP IPL command.
- The user (either manually or through an EXEC procedure) issues the CP CLOSE command to close the spool file.
- VS1 with handshaking uses the diagnose interface to issue the CP CLOSE command after each job completes.
- The installation modifies the operating system by adding DIAGNOSE instructions to communicate with VM/SP to close the spool files.

Thus, until one of the preceding actions occur, the spool files are not sent to the real printer or punch. To keep spool files from building up excessively on the spooling DASD, the user should close these spool files periodically (such as at the end of each job).

PROCESSOR MODEL AND CHANNEL MODEL DEPENDENCIES

Channel checks (that is, channel data checks, channel control checks, and interface control checks) no longer cause the virtual machine to be reset. Thus, an operating system that now runs in a virtual machine can attempt either to recover from a channel check or to terminate in an orderly manner.

If channel error recovery procedures in an operating system depend on the processor model and channel model, then these two requirements must be met for channel error recovery procedures to function reliably after a channel check:

1. Depending upon the recovery procedures of the specific operating system running in the virtual machine, an installation may have to generate the operating system for the same processor model on which VM/SP is to run.
2. The virtual machine configuration must have each virtual channel correspond to a single type and model of real channel.

The second requirement means that all virtual devices on a virtual channel must correspond to real devices on real channels; the real channels must be identical to each other in type and model. For example: Assume that a virtual machine has a 3330 disk on virtual address 280 and a 3340 disk on virtual address 290 that correspond to similar real devices on real addresses 380 and 590, respectively. Because both virtual devices (280 and 290) are on a single virtual channel (channel 2), the corresponding real devices (380 and 590) must both be on real channels that have an identical channel type and model. By meeting this requirement, when an operating system issues a STIDC (store channel ID) instruction to virtual channel 2, VM/SP can simulate it the same way and return consistent results to the operating system.

Not only should the real channels be identical, but generally speaking, they should be of the same type as the virtual channel. (The virtual channel type is defined either in the OPTION statement in the virtual machine's directory entry or by the class G CP DEFINE CHANNEL command.) Two exceptions to this general rule are:

- When the real channel is a block multiplexer channel, the virtual channel can be a selector channel. In this case, the simulated STIDC instruction returns this information to the operating system: (1) the model number of the block multiplexer channel, and (2) a channel type field indicating that the channel is operating in selector mode.
- When the real channels are selector channels, the virtual channel can be a block multiplexer channel. This specification may improve performance when the virtual channel has devices on several real selector channels. It allows the virtual machine to overlap channel operations on the virtual channel and to take full advantage of the several selector channels. However, when VM/SP simulates the STIDC instruction issued to the virtual block multiplexer channel, it gives the operating system the channel type and model number of a selector channel, not of a block multiplexer channel. While this result is inconsistent with the channel's operation as a block multiplexer, the operating system should not detect, or be affected by, this inconsistency.

For further restrictions about channel model-dependent functions, refer to VM/SP Planning and System Generation Guide.

Specifying a Secondary Userid

Normally, a disconnected user has no console services. However, by specifying a secondary userid on the CONSOLE directory control statement, a virtual machine can retain console services while disconnected. Any console output created by the disconnected virtual machine or CP on behalf of the disconnected virtual machine transfers to the console of the secondary user. Also, using the CP SEND command, the secondary user can issue commands and replies on behalf of the disconnected user. For further information on the SEND command, refer to VM/SP CP Command Reference for General Users.

Note: In order to use this secondary user support, the virtual console of the disconnected user must be used as a 3215 console. Full-screen-display type operations are rejected, causing a "command reject" situation. Therefore, the disconnecting virtual machine must be out of full screen mode before issuing the CP DISCONNECT command.

SPECIFYING VIRTUAL MACHINE CONSOLES

To specify more than one console for a virtual machine, the virtual machine user must tell VM/SP about the existence of these additional consoles. Operating systems may support either:

- Multiple consoles -- where different classes of system messages can be routed to different consoles.

--or--

- Alternate consoles -- where the user can switch to a backup console when the primary console becomes inoperative.

To tell VM/SP about the existence of these consoles, either use directory statements or issue CP commands. The way a user specifies the second console depends upon whether:

- The user always wants to use a specific device at a specific I/O address.

--or--

- The user wants flexibility in selecting which device or terminal is to be used.

Using Specific Devices as Virtual Consoles

Assumption: An OS/VS virtual machine is to run its 3158 display console at address 01F in display operator console mode. Also, the virtual machine is to operate a local 3270 terminal at address 1B8.

Step 1: Generate OS/VS with at least two consoles: 01F as the primary console, and 009 as the secondary console.

Step 2: Specify the secondary console by using the VM/SP CONSOLE directory statement. Code it:

```
CONSOLE 009 3210
```

Step 3: Specify the OS/VS primary console either by having a DEDICATE statement in the VM/SP directory or by using the CP ATTACH command after logon. Either specification allows OS/VS to use the 3158 console in display operator console mode.

If the DEDICATE statement is used, then code it:

```
DEDICATE 01F 01F
```

If the ATTACH command is used, then, after logon, send a message to the VM/SP operator that requests the following ATTACH command to be issued:

```
msg op attach 01f to userid as 01f
```

Using a Display Terminal as a Console in Display Mode

To use any locally attached 3270 display terminal as the OS/VS or DOS/VS primary console in display operator console mode, either have a SPECIAL statement in the virtual machine's VM/SP directory entry or issue the CP DEFINE command after logon to VM/SP.

If the SPECIAL statement is used, it appears as follows:

```
SPECIAL 01F 3270
```

If the SPECIAL statement is not used, assume that a local 3270 line has been enabled by the VM/SP operator. Then, issue the following DEFINE command:

```
define graf 01f 3270
```

In either situation, after a user logs onto VM/SP (by using the device specified in the CONSOLE statement) and loads the operating system into the virtual machine (by using the IPL command with the STOP option), a user must issue the CP DIAL command at the local 3270 that is to be used in display mode. This action logically connects that 3270 and its real I/O address of 1B8 to the operating system. To drop the dialed connection, a user must issue the CP RESET command.

Note: A remote 3270 cannot be used in this manner because the DIAL command does not support remote 3270 terminals.

If the second console is a remote terminal such as a 2741 or 3767 connected by either a 2702 or a 3704/3705 in 2702 emulation mode, the SPECIAL statement would appear as follows:

```
SPECIAL 01F 2702 IBM
```

The DEFINE command would be:

```
define line as 01f ibm
```

VIRTUAL MACHINE I/O MANAGEMENT

A real disk device and a real or emulated 270x transmission control unit (TCU) can be shared among multiple virtual machines. Virtual disk device sharing is specified in the VM/SP directory entry or by a user command. A specific virtual machine may be assigned read-only or read/write access to a shared disk device. To gain access to the shared virtual device, a user must supply the appropriate password. To ensure device integrity, VM/SP checks each virtual machine I/O operation against the parameters in the virtual machine configuration.

The virtual machine operating system is responsible for the operation of all virtual devices associated with it. These virtual devices may be defined in the VM/SP directory entry of the virtual machine, or they may be attached dynamically to (or detached from) the virtual machine's configuration for the duration of the terminal session.

Virtual devices may be in one of these three states or conditions:

- Dedicated -- when mapped to a fully equivalent real device
- Shared -- when mapped to a minidisk or when specified as a shared virtual disk device
- Spooled -- when VM/SP places the device output on intermediate direct access storage

For example: In a real machine when running under control of the operating system, the problem program requests the system to issue a SIO instruction to a specific device. The operating system normally initiates the I/O operation and handles any device error recovery. In a virtual machine, the operating system performs the same functions, but the device address specified and the storage locations referenced are both virtual. VM/SP has the responsibility for translating the virtual specifications to real.

When I/O interruptions occur, VM/SP reflects them to the virtual machine for its interpretation and processing. When I/O errors occur, VM/SP records the error, but it does not initiate error recovery operations. The operating system must handle error recovery.

When VM/SP initiates I/O operations for its own paging and spooling, the operation is not subject to translation, and VM/SP itself performs the operation.

Dedicated Channels

In most cases, many virtual machines share the I/O devices and control units on a channel both as minidisks and dedicated devices and with VM/SP system functions such as paging and spooling. Because of this sharing, VM/SP has to schedule all the I/O requests to achieve a balance among virtual machines. In addition, VM/SP must reflect the results of the subsequent I/O interruption to the appropriate storage areas of each virtual machine.

By specifying a dedicated channel for a virtual machine (using the class B ATTACH CHANNEL command), the virtual machine has the channel and all the devices for its own exclusive use. For dedicated channels, VM/SP translates the virtual storage locations specified in channel commands to real locations and performs any necessary paging operations. However, VM/SP does not translate a device address because the virtual device addresses on the dedicated channel must match the real device addresses; thus, minidisks cannot be used.

Dedicated devices should be considered as an alternative to dedicated channels because then the only translation done by VM/SP is device address translation. Dedicated devices should be specified on separate channels so that VM/SP can handle them more efficiently for a virtual machine.

Defining Direct Access Storage Devices

Virtual machines can use DASD as either minidisks or dedicated volumes. A real disk volume can be shared by several virtual machine users, each owning a number of contiguous cylinders or blocks. This logical subdividing of a real disk volume is called physical pack sharing, and each subdivision of cylinders or blocks is called a virtual disk or minidisk. A real disk volume can also be dedicated to a specific virtual machine for its own private use. When using dedicated disk volumes, the virtual machine's operating system must perform all necessary interruption handling, error recovery, and error recording.

By using either the LINK directory control statement or the CP LINK command, a user can share the data on a minidisk or entire disk volume with the owner of the virtual disk. The LINK statement or command allows controlled, concurrent access to the data on the virtual disk. This sharing is called logical data sharing.

If any virtual machine temporarily requires additional direct access space, the user can use the CP DEFINE command to obtain it dynamically from a pool of temporary (T-disk) space. To define a pool of T-disk space, an installation specifies the size of the T-disk pool when allocating disk space with the stand-alone CP format/allocate program.

Before using the T-disk space, the user must first initialize it. For storing DOS, OS, or VSAM files, use the IBCDASDI program to initialize the minidisk and set up the VTOC. For CMS files, issue the CMS FORMAT command. Temporary minidisks are available to the virtual machine for the duration of the current terminal session. The area is returned to VM/SP when one of the following actions occur:

- The system forces the virtual machine off.
- The virtual machine logs off.
- The user issues a CP DETACH command to release the temporary minidisk.

For details about defining, formatting, using, and sharing minidisks, refer to VM/SP Planning and System Generation Guide.

Note: The IBCDASDI program cannot be used to format disk space on a 3380. If a guest operating system requires disk space on a 3380, refer to the Device Support Facilities manual Order No. GC35-0033. This manual describes the procedures for initializing, formatting, and analyzing disk space on a 3380.

VM/SP Alternate Path Support

VM/SP alternate path support permits the definition of alternate paths to a tape or DASD unit on the VM/SP processor; this option supports the two/four channel switch and string switch features. Define alternate paths in VM/SP's DMKRIO for devices that a virtual operating system is to use. When you do, VM/SP will map I/O requests from a virtual address associated with the virtual machine to one of the real paths to the device as defined in DMKRIO. Refer to the section, "VM/SP Alternate Path Support", in the VM/SP Planning and System Generation Guide for an explanation on the definition of alternate paths.

As a rule, alternate path and reserve/release support are mutually exclusive. There is, however, one exception to this rule. At the minidisk level, VM/SP provides virtual reserve/release support in the form of a software locking mechanism. As long as virtual machines under VM/SP use virtual reserve/release between them and as long as other real processors do not share the volume, alternate paths can be defined for the real device. Refer to the section, "Operating Systems Using DASD Reserve/Release" for specific information on DASD sharing and virtual reserve/release.

For specific information on channel switching, refer to the VM/SP Planning and System Generation Guide.

OPERATING SYSTEMS USING DASD RESERVE/RELEASE

Reserve/release CCW commands prevent several users of the same data files from simultaneously accessing the same data. It is most useful when that data is being updated. While VM/SP handles the reserve/release CCW commands presented by other operating systems running in virtual machines, VM/SP and CMS do not use reserve/release CCW commands. Operating systems use these commands under two conditions:

- When running in virtual machines under VM/SP and sharing data files
- When running on other processors and sharing data files with operating systems that run under VM/SP

VM/SP has two types of reserve/release support:

- Shared DASD -- applies to virtual machine operating systems that issue reserve/release CCWs to preserve data integrity; the hardware reserve preserves the data integrity on a device basis.
- Virtual Reserve/Release -- applies to virtual machines issuing reserve/release CCW commands to minidisks that are designated as subject to virtual reserve/release processing.

SHARED DASD

Reserve/release support allows several operating systems, such as MVS, SVS, and VS1, whether running as virtual machines under VM/SP or on other processors to have data protection on a full volume. The hardware reserves the device when a reserve CCW command is executed. VM/SP supports reserve/release CCW commands for shared DASD as though each virtual machine has a separate channel path to a shared device.

Note: When a reserve is issued to a device that has alternate path support (defined in the RDEVICE and RCTLUNIT VM/SP system generation macro instructions), VM/SP changes a reserve CCW command to a sense CCW command.

Virtual Reserve/Release Support

Virtual reserve/release support allows several operating systems, such as MVS, SVS, and VS1, to all run as virtual machines under the same VM/SP operating system and to have data protection when using the same data files on the same minidisk.

To use virtual reserve/release, specify "V" in the mode operand of the MDISK directory statement. Also subject to virtual reserve/release processing are the virtual machine users who use the same minidisk by way of LINK statements.

By using the VM/SP virtual reserve/release support, one operating system running in a virtual machine can prevent other operating systems running under the same VM/SP system from accessing the reserved minidisk. However, a minidisk protected by virtual reserve/release support may not be protected from access by an operating system running on other processors.

Restrictions: Device Sharing Between Real Processors

- When a device is shared between processors and at least one of the processors is running VM/SP, the shared volume cannot contain more than one minidisk. The single minidisk may encompass the entire volume or a small portion of the volume. Neither CP nor any virtual machine may reference the remainder of the volume for use as paging, spooling, etc. device.
- Devices shared between processors must not be generated in VM/SP's DMKRIO as having alternate paths. It may happen that there are multiple paths from the VM/SP processor to the shared devices and a path from these shared devices to another processor. If so, you may not use the ALTCH or ALTCU macro instruction operands to define as alternate the paths from the VM/SP processor. This means that the definition of alternate paths in DMKRIO and the use of real reserve/release are mutually exclusive.

Restrictions: Device/Minidisk Sharing On a Single Processor

- If more than a single path to a volume exists, DMKRIO may be generated so that each path is defined as a separate path, not as an alternate path. When this is done, each path can be attached or dedicated to a different user, and reserve/release CCWs issued by such users preserve the data integrity. In this case, the integrity is preserved by the hardware, not by the software reserve/release support. Again the definition of alternate paths in DMKRIO and the use of real reserve/release are mutually exclusive.
- A volume may be defined through the directory to contain one or more minidisks. Such minidisks must be identified through the MDISK statement as requesting virtual reserve/release support. These minidisks may then be shared between virtual machines that support Shared DASD (not CMS) and the data integrity will be preserved by the use of reserve/release CCWs in the virtual machine channel program. Alternate paths may be defined to the device when using virtual reserve/release. The reserve CCW will still be changed to a sense CCW but the integrity will be preserved by the virtual reserve/release code.

Example -- Reserve/Release for Dedicated Volumes

In this example, 230 and 330 are alternate device addresses for a particular DASD to be shared by USERA and USERB (two virtual machines running on the same real computing system that supports sharing). To share this device:

1. Generate the virtual machine operating system for USERA to support both the device at 230 (two-channel switch) and the reserve/release software.
2. Generate the virtual machine operating system for USERB to support both the device at 330 (two-channel switch) and the reserve/release software.
3. Generate VM/SP as though 230 and 330 were different devices (with different control units and channels).
4. Issue the CP ATTACH command to attach device 230 to USERA and device 330 to USERB.

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

- Generate the VM/SP system with device 230 but not 330.
- Issue the CP ATTACH command to attach device 230 to USERA.

In both cases:

- The device addresses generated for systems to run in a virtual machine need not be the same as on the real machine.
- The devices used by virtual machines must be dedicated (attached or defined with a DEDICATE statement in the VM/SP directory).

Do not share the CP SYSRES and any other CP-owned disk between two processors.

Summary of Reserve/Release Support

VM/SP checks all CCW commands passed by operating systems running in virtual machines. It bases reserve/release CCW command processing on: the type of device, the presence or lack of alternate path support, and whether the MDISK statement in the VM/SP directory contains a "V" on the mode operand. For the hardware to execute the reserve/release CCW commands, the two-channel switch special feature must have been installed. Depending upon the various combinations of these items, VM/SP either permits the reserve CCW command to execute on the hardware or changes the reserve CCW command to a sense CCW command. To determine the conditions when a "reserve" is changed to a "sense" CCW command, refer to Figure 1.

Type of Device	Alternate Path Support	Reserve/Release Executes in the Hardware (2-4 Channel Switch)	Virtual Reserve/Release Requested (V Added to Mode in MDISK)	CCW Comnd Sent by VM/SP to Device	Note
Dedicated DASD or Tape	Not defined	Not applicable	Not applicable	Reserve	1
	Defined	Not applicable	Not applicable	Sense	2
Minidisks	Not defined	Yes	No	Reserve	1
	Not defined	Yes	Yes	Reserve	1
	Not defined	No	No	Reserve	3
	Not defined	No	Yes	Sense	4
	Defined	Not applicable	Not applicable	Sense	2

¹Normal Operation -- The command is passed unchanged to the hardware.

²When the VM/SP system has been generated with alternate path support for those devices, it prevents the devices from being reserved. This action causes VM/SP to avoid a possible channel lockout. VM/SP does not return any indication of this action to the operating system issuing the CCW command that the device was not reserved.

³Without the two-channel switch special feature, VM/SP sends the reserve/release CCW command unchanged to the hardware. However, the hardware rejects the command and does not reserve the device.

⁴Before sending the command to the hardware, VM/SP changes the reserve CCW command to a sense CCW command and places a virtual reserve on the minidisk. The real device is not reserved. The virtual reserve prevents other operating systems running under the same VM/SP system from accessing the minidisk; however, these same virtual operating systems may reserve other minidisks located on the same real volume. Because the two-channel switch feature is not installed on the channels, only one address path goes to the device from the VM/SP processor. This path allows VM/SP virtual reserve/release processing to send a sense CCW to the device, although the reserve CCW command would be rejected by the hardware.

Figure 1. Summary of VM/SP Reserve/Release Support

FULL SCREEN CONSOLE SUPPORT FOR VIRTUAL OPERATING SYSTEMS

Prior to VM/SP, a guest operating system running in a virtual machine was forced to attach a local 3270 display device in order to utilize the full screen console support available within that guest system. Unfortunately, this method created several disadvantages:

- The attached terminal could not enter CP commands.
- The attached terminal could not display CP messages.
- The logon terminal was only needed to overcome the two previous problems.

VM/SP, however, provides full screen console support for virtual machine operating systems that can take advantage of the full screen mode of operation. This support removes the need to attach an additional 3270 terminal. A user can logon a local 3270 display and alternate between full screen mode (graphics device) and CP mode (line device).

A virtual console operator determines when to switch between CP mode and full screen mode. In addition, the virtual console operator has the option of saving the full screen image before switching to CP mode. Upon returning to full screen mode, the saved screen image is restored. For a further description of mode switching, refer to the CP TERMINAL command in VM/SP CP Command Reference for General Users.

Switching Modes in a Virtual Machine Environment

If the virtual operating system needs a printer, the virtual console operator can perform the following steps:

1. Enter CP mode (via the break-in key).
2. Ask the operator to dedicate a printer to the virtual machine.

```
msg op please attach the 3800 printer as 001
```
3. Wait for the operator to answer your request.

```
sleep
```

```
PRINTER 001 ATTACHED
```
4. Return to full screen mode (type BEGIN).
5. Initialize and start the printer so the virtual machine operating system can take advantage of it.

With this new facility in VM/SP, an installation does not have to sacrifice two terminals in order to run an operating system that relies on full screen support. For a detailed description of full screen console support, see the VM/SP System Programmer's Guide.

Notes:

1. Although 3270 terminals with different screen sizes are supported, the virtual operating system is responsible for issuing the correct 3270 start I/O commands.
2. Although the virtual operating system can use a local logon 3270 terminal for full screen support, the guest cannot get an attention interrupt from the break-in key since CP uses it for mode switching.
3. Spooled console files contain no information entered while in full screen (3270) mode. However, this problem can be solved by console spooling under control of the guest operating system. This avoids the loss of any information.
4. In order to use CMS, the console must be in line (3215) mode. In fact, issuing the IPL CMS command automatically resets the console to line mode.

ALTERNATING BETWEEN OPERATING SYSTEMS

A virtual machine user may require the facilities of more than one operating system during a single terminal session.

When running an operating system from a terminal, use the CMS editor to create and modify job streams and to analyze the results and output.

Application programmers who normally use CMS to interactively create, modify, and test programs, may require facilities for compilation or execution that are not supported or available in CMS.

The technique described in this topic uses multiple operating systems consecutively. Job control cards, compiler or assembler source programs, and test data streams are created and modified at the terminal under control of the CMS editor. The job stream is then executed, by passing control to an appropriate operating system that has the necessary facilities.

In this way, the programmer uses the terminal-oriented facilities of CMS to create and update source programs and JCL. When ready to compile or test, he can give control of his virtual machine to the operating system. After execution is finished, he can give control back to CMS to selectively scan and display printer and punch output at the terminal.

This approach assumes that the programmer has created source program files and data files under CMS. To execute under another operating system (in this example, OS), the programmer must also create JCL records that specify the compilation, link editing, or execution, as appropriate. These records are created under CMS and named with a distinctive filename and filetype (for example, PLICOMP JCL). Job control records, source program files, and data files can then be merged together in the virtual card reader to form a single OS job stream. The CP and CMS commands (shown in Figure 2) create and transfer this job stream.

Transferring Output

The CP SPOOL command transfers subsequent (not currently existing) card images from the virtual card punch of one virtual machine to the virtual card reader of that same or some other virtual machine. During this time, no real cards are punched or read; VM/SP manages the transfer of CMS card-image data files through disk spooling operations only.

Figure 2 shows how to punch files to the virtual machine's card reader. The virtual machine is in the CMS environment at the start of the example. The command "SPOOL 00c cont eof" specifies that reading be continuous until all files spooled to the virtual machine are exhausted and the virtual end-of-file button on the reader is pushed. NOHEADER specifies that no special control cards are to be inserted at the beginning of each punched file. Virtual device 230 is an OS system volume. Virtual device 231 contains the OS job queue, SYS1.SYSJOBQE. All standard CMS and OS responses are omitted from the example; however, the OS READY message is included to more fully illustrate the IPL sequence. Also, assuming that the user has a 2741, he must press the attention key before entering each OS command. The attention interruptions are not shown in Figure 2.

```

| CMS
| cp close 00c
| cp purge 00c all
| cp clcse 00d purge
| cp spool 00d to * cont
| punch jobcard jcl (noheader)
| punch plicomp jcl (noheader)
| punch plimain pli (noheader)
| punch asmcomp tcl (noheader)
| punch asmsub assemble (noheader)
| punch linkgo jcl (noheader)
| punch qodata dat (noheader)
| punch slshstar jcl (noheader)
| cp spool 00d nocont
| cp close 00d
| cp spool 00c cont eof
| cp ipl 230
|
| Note: The following are issued once under OS control:
|
| IEE007A READY
| set date=xx.355,Q=(231)
| start rdr,00c
| start wtr,00e
| start

```

Figure 2. OS Job Stream Transfer

To transfer files between systems, the user must have access to both operating systems being used. Access to both systems can be provided either in the virtual machine's VM/SP directory entry, or dynamically before loading the new system.

Figure 3 illustrates a virtual machine configuration and the corresponding VM/SP directory control statements. Virtual device addresses 190 and 191 contain the CMS system and user disk area. Virtual device addresses 230 and 231 contain the OS system and user disk area. The two systems use a common card reader, card punch, printer, and console.

```

| USER CS2 PASSWORD
| ACCOUNT NUMBER BIN16
| CONSOLE 01F 3215
| SPOOL C 2540 READER
| SPOOL D 2540 PUNCH
| SPOOL E 1403
| LINK JFK 230 230 R
| LINK CMSSYS 190 190 RR
| MDISK 231 2314 120 82 UDISK1 WR
| MDISK 191 2314 101 10 UDISK1 WR RPASS WPASS

```

Figure 3. Directory Entry for Alternating Between Operating Systems

Configurations for Alternating Between Operating Systems

Users can alternate between operating systems more simply if:

- Devices used by both systems are supported at the same device address.

--and--

- Common addresses are not used to support different devices.

If these two conditions are not met, the user must modify the virtual machine configuration before each IPL of a new system.

If the two systems require online typewriter keyboards at different addresses, use the CP DEFINE command to change the address of the virtual system console. For example: The OS system (specified above) requires an online typewriter keyboard at address 01F, while the CMS system probably has its console at address 009. In this case, issue this command before loading 230:

```
cp define 009 as 01f
```

Because CMS automatically communicates with any valid multiplexer address, a user can leave the console at 01F and satisfy both systems.

If the systems expect different device types at the same address, the common address must be assigned to the appropriate device each time a new system is loaded. If CMS is running with a disk at address 191 and OS is generated to support a 3330 at that address, issue the following command before loading OS:

```
cp detach 191
```

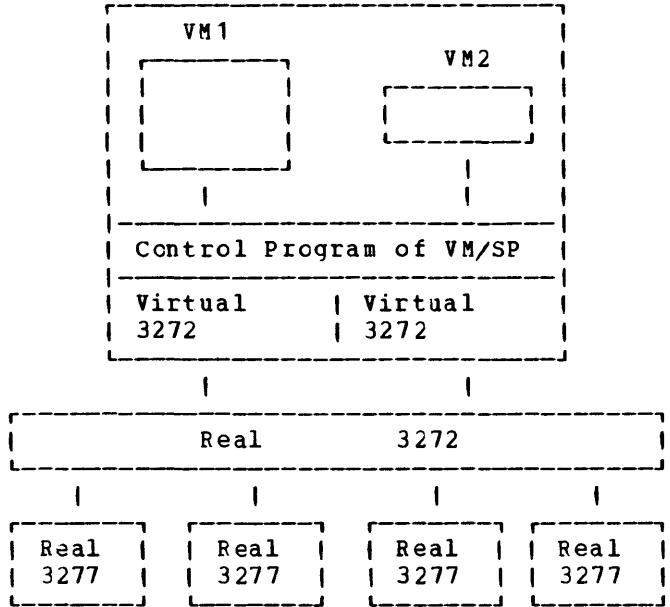
An appropriate device can then be added to the virtual machine at address 191. Add the device either before loading or in response to a mount request from the OS system.

Note: For direct access storage devices, this procedure is necessary even if both systems support the same device type at the same address. Except for VSAM disks, the disk format used by CMS is unique. It is not compatible with that of other operating systems. Files can be passed between CMS and OS or DOS only through VM/SP spooling or through VSAM data sets.

MULTIPLE-ACCESS VIRTUAL MACHINES

Multiple-access programs execute in a virtual machine and directly control terminals. These terminals do not have to be supported by VM/SP as virtual operator consoles, but they may be of any type supported by the program executing. These programs use lines that are either dedicated to the virtual machine (by the directory entry) or assigned to the virtual machine dynamically.

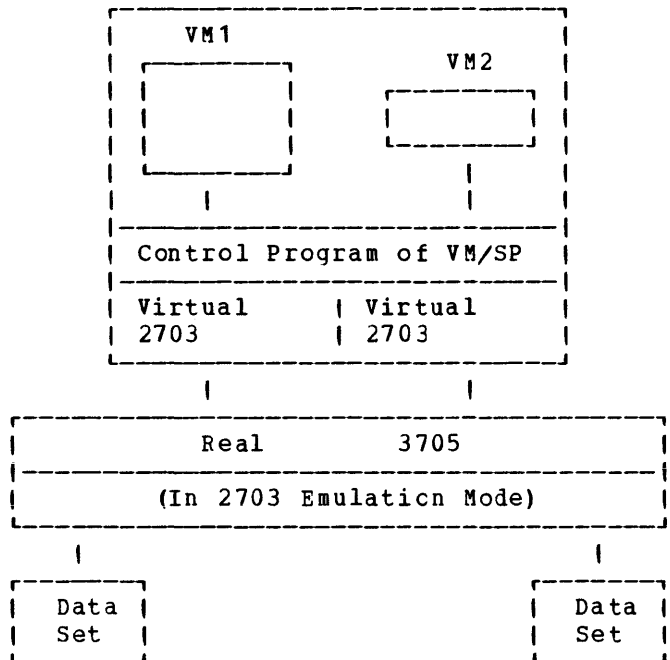
For example: Figure 4 shows two multiple-access systems (controlled by virtual machines VM1 and VM2). While each system controls real 3277s by using part of the real 3272, the real 3272 appears to both virtual machines as though they each have sole control of it. (The virtual system consoles of VM1 and VM2 are not shown.)



Note: Users can define virtual lines for a virtual machine. These lines are a subset of the lines controlled by a real transmission control unit (TCU).

Figure 4. Virtual Devices: Local 3270 Terminals

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



Note: Two lines on the real 3705 are defined as virtual lines for two virtual machines named VM1 and VM2. The remaining lines may support virtual operator consoles.

Figure 5. Virtual Devices: Remote Terminals

As shown in Figure 6, the virtual machine operating system may be one like VM/SP itself, or TSO, that supports a number of remote terminals.

To assign a real line as a virtual line, the terminals supported by the virtual machine's operating system are of the same type as those supported by VM/SP as virtual system consoles. To make this assignment, define the virtual lines either in the virtual machine's VM/SP directory entry (via the SPECIAL control statement) or add them to the logged-on virtual machine (via the CP DEFINE command).

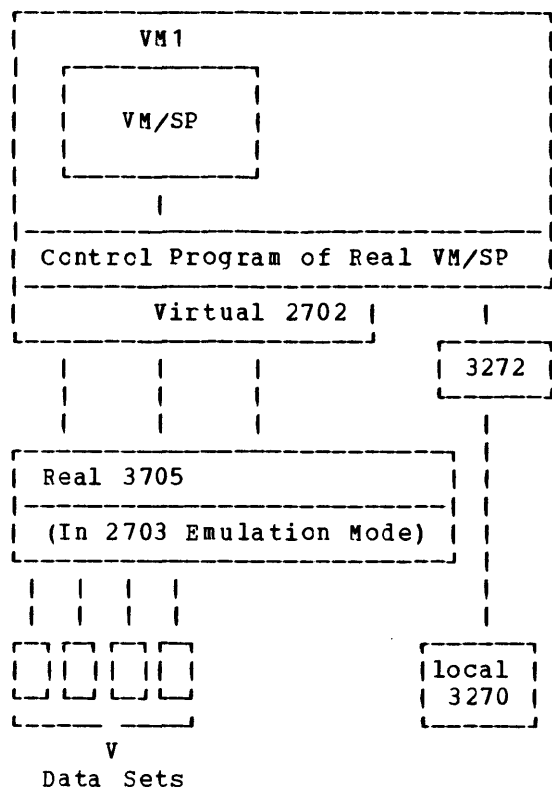


Figure 6. A Virtual VM/SP Multiple-Access System

Figure 7 illustrates a VM/SP directory entry for a multiple-access virtual machine to run VM/SP under VM/SP.

```

USER TESTVM PASSWORD 1M
OPTION REALTIMER ECMODE
CONSOLE 01F 3215
SPOOL 00C 2540 READER
SPOOL 00D 2540 PUNCH B
SPOOL 00E 1403 A
DEDICATE 190 SYSRES
DEDICATE 191 SYSWRK
SPECIAL 080 2702 IBM
SPECIAL 081 2702 IBM
SPECIAL 082 2702 IBM
SPECIAL 083 2702 IBM
SPECIAL 070 3270
  
```

Figure 7. Directory Entry for a Multiple-Access Virtual Machine Running VM/SP under VM/SP

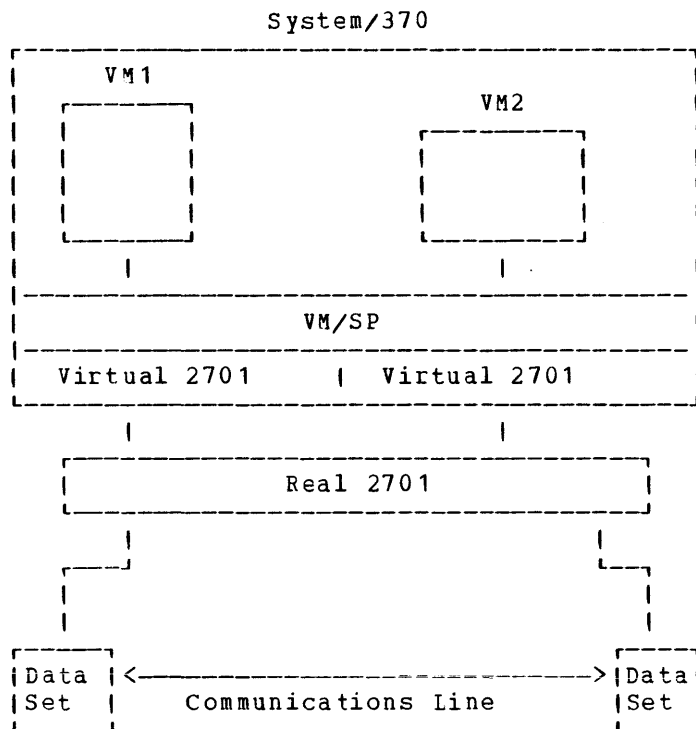
To connect a terminal supported by both VM/SP and a multiple-access system, use the CP DIAL command. Such terminals can be on either non-switched or switched lines. To connect a terminal to the virtual machine defined in Figure 7, enter this command:

```
dial testvm
```

The VM/SP system matches the terminal type to an equivalent virtual line that is available and enabled (in this example, 070, 080, 081, 082, or 083). Once a connection is made, the virtual machine controls the terminal to which it is logically connected (in this example, the VM/SP virtual machine). It remains connected until logoff using standard logoff procedures or until the virtual machine is forcibly logged off. Once logged off, the user is then free either to logon to VM/SP or to use the DIAL command to contact another multiple-access system.

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

As shown in Figure 8, a communications system can be tested by using multiple virtual machines in place of multiple real machines. For example: While there exists a single two-line 2701 on the real machine, the virtual 2701 units could each be defined as a one-line 2701.



Note: This figure assumes that the real 2701 transmission control unit is equipped with the appropriate data sets and line capability.

Figure 8. A Communications Test System

Figure 9 illustrates a virtual transmission control unit running remote 3270 units.

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

- Defined in the VM/SP directory entry for the virtual machine via the DEDICATE control statement; for example:

```
DEDICATE vaddr raddr
```

where: vaddr is the virtual address, and raddr is the real address of the appropriate line appearance on the real transmission control unit.

-- or --

- Attached to the virtual machine by an operator with privilege class B; for example:

```
attach raddr to vm1 as vaddr
```

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

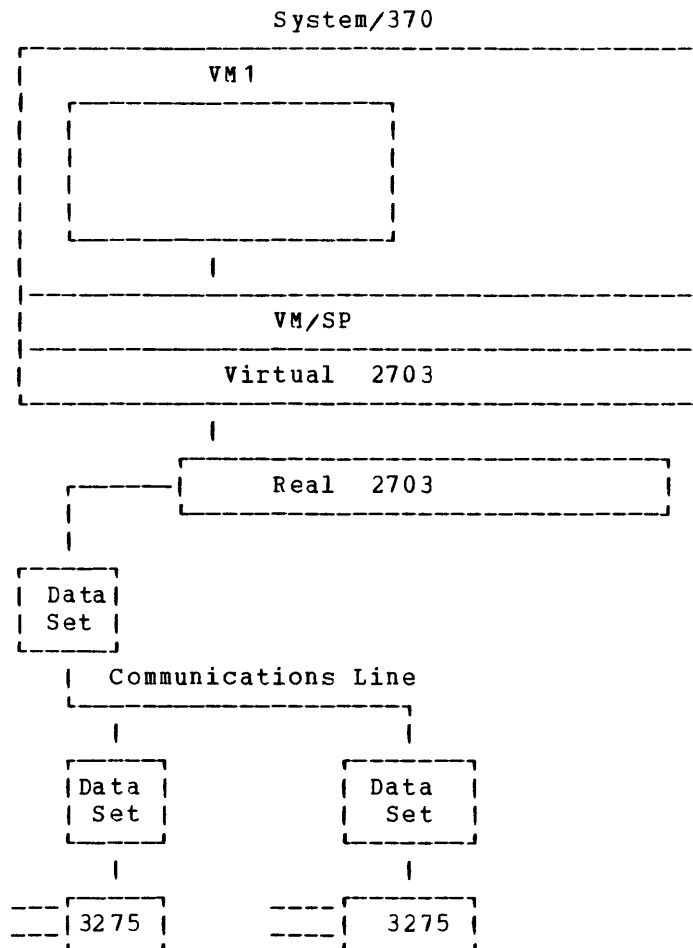


Figure 9. A Virtual 2703 TCU Controlling Remote 3270 Terminals

Performance Considerations

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

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

If the program must be run in the dynamic paging area, then locking specific pages of the virtual machine into real storage may ease the problem. However, besides page zero and the page containing the teleprocessing interruption handler, it is not always easy or possible to identify which specific pages are always required.

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

For details about the locked pages and reserved page frames options, refer to the VM/SP System Programmer's Guide.

THE ASP VIRTUAL MACHINE

When using the OS asymmetric multiprocessing system (ASP), an installation may find virtual machines useful in two ways.

The first way, as shown in Figure 10, has a virtual ASP system that can be run by two virtual machines (VM1 and VM2) using a virtual channel-to-channel adapter (CTCA).

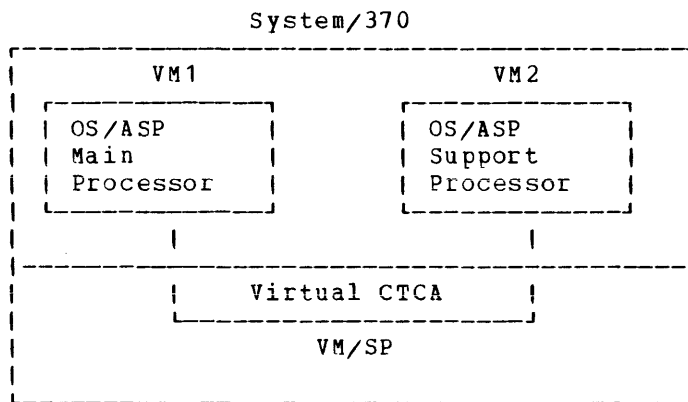


Figure 10. Two ASP Virtual Machines

The virtual ASP system (VM1 and VM2) shown in Figure 10 may be used in one of two ways: (1) to install and test a new ASP release, or (2) for ASP system testing in a virtual environment concurrent with normal production. The virtual ASP system eliminates the need to dedicate the real ASP system for new system testing.

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

```
SPECIAL 280 CTCA
```

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

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

```
couple 280 to vm2 380
```

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

The second way, as shown in Figure 11, has an additional virtual machine (VM3), with a real channel-to-channel adapter to a real System/360 or System/370, running as either the main or support processor in a production ASP system.

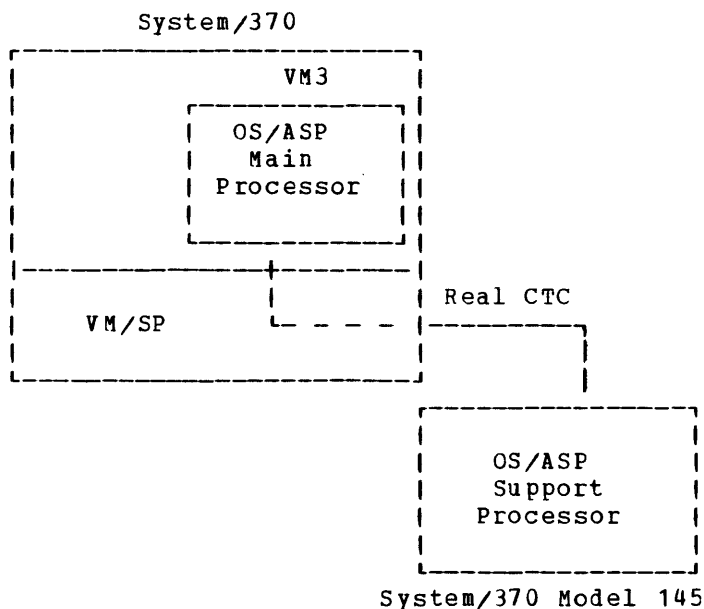


Figure 11. One Real and One Virtual ASP Machine

Define the real channel-to-channel adapter in the VM/SP system generation procedure by using the RDEVICE macro instruction with a device type of CTCA (DEVTYPE=CTCA). The virtual machine (VM3) must have this device assigned to it before the IPL. Make this assignment by using the DEDICATE statement in the virtual machine's VM/SP directory entry, such as:

```
DEDICATE 280 380
```

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

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

Note: For further information on the virtual channel-to-channel adapter, refer to the description of the COUPLE, DEFINE, and DETACH commands in the VM/SP CP Command Reference for General Users.

Shadow Table Maintenance Support

Shadow table maintenance support reduces the overhead associated with maintaining shadow page, and segment tables. This support includes the following four areas.

- Multiple shadow table support
- Selective invalidation
- Shadow table bypass for V=R users
- Shadow table bypass for V=V users

MULTIPLE SHADOW TABLE SUPPORT

Multiple shadow table support reduces the number of shadow tables that VM/SP has to purge when a guest operating system in a virtual machine dispatches a new address space. This support adds a segment table origin control block (STOBLOK) and the SIMULTI n option to the SET command.

STOBLOK keeps all the information about each shadow segment table, and VM/SP maintains a queue of STOBLOK control blocks and associated shadow tables for each guest EC mode virtual machine. Thus, each time a guest operating system dispatches a new address space via a LCTL instruction, VM/SP can locate the proper shadow table to be used.

By issuing the SET STMULTI n command, a user can define how many shadow tables that VM/SP should maintain concurrently for each virtual machine. The maximum number is six, and the default is three. The actual number specified varies by the amount of free storage available. Each segment table for a 16 megabyte address space requires 1024 bytes of storage, plus space for the page tables. To display the SIMULTI specifications, a user can issue the QUERY SET command.

SELECTIVE INVALIDATION

Selective invalidation is a standard function of shadow table maintenance support. It allows VM/SP to selectively invalidate a shadow page table entry when a page frame is stolen or released from a guest EC mode virtual machine. Selective invalidation always takes place below the high-water mark, which is established with the SET STBYPASS nnnnk command. Above the high-water mark, selective invalidation occurs only if virtual machine assist is off. When virtual machine assist is on, the shadow page table entries above the high-water mark are invalidated whenever a page frame is released or stolen. After the guest EC mode virtual machine causes a page fault, virtual machine assist revalidates those entries above the high-water mark.

SHADOW TABLE BYPASS FOR V=R USERS

Shadow table bypass for V=R users eliminates shadow tables for guest operating systems executing in the V=R virtual machine. By issuing the SET STBYPASS VR command, a user eliminates shadow tables and the overhead associated with maintaining them. VM/SP modifies the virtual operating system's page table to relocate virtual page zero to the highest real address within the V=R area. This relocation makes it possible for VM/SP to dispatch the virtual machine and have real control register 1 point to the guest page and segment tables.

Note: Single processor mode requires the use of shadow tables to simulate virtual prefixing. The SET STBYPASS VR command is ignored if issued. However, the SET STBYPASS nnnnk command is valid and should be used in the single processor mode environment.

SET STMULTI Command

If the SET STBYPASS VR command has been issued and shadow tables have been eliminated, the SET STMULTI command has no effect on the V=R guest virtual machine. However, the single processor mode V=R user running a guest AP or MP system can use effectively the SET STMULTI command.

Restrictions

When shadow tables are eliminated, the following restrictions apply:

- The virtual system's real page zero must map only to its virtual page zero. Otherwise, STBYPASS VR will be set off and shadow tables will be used instead.
- No virtual machine segment or page tables can start in a relocated page. This means that VM/SP control register 1 and the segment table entries cannot point to the first 4K of storage.
- The system cannot use the relocated page table entry in these ways:

--By looking at its contents

--or--

--By executing a load real address (LRA) instruction on virtual page zero (normally mapped to real page zero), except for using the condition code returned by the LRA instruction

- The virtual operating system must have only one page table entry for its real page zero. If multiple address spaces are used, the page table must be shared by each address space that uses real page zero.

Note: Once relocated by the SET STBYPASS VR command, the virtual operating system must continue to use the relocated page table entry without changing its contents or moving its location.

- Any dump taken of the virtual operating system may contain a relocated page table entry for page zero. Thus, any program designed to automatically read and interpret dumps must handle this condition.

SHADOW TABLE BYPASS FOR V=V USERS

V=V users executing SVS, MVS, or VM/SP under VM/SP can use shadow table bypass. This function allows V=V users to establish a common nucleus for multiple address spaces. By specifying the maximum size of the nucleus, a user reduces purge and invalidation time. Thus, when the virtual machine executes a PTLB or LCTL instruction, VM/SP invalidates or purges only the shadow table entries that are above the high-water mark (or highest limit) of the virtual operating system's nucleus that is mapped guest virtual=guest real.

SET STBYPASS Command

By using the SET STBYPASS nnnnnK command (in conjunction with the STFIRST directory option), a user can define the high-water mark (or highest limit) of the virtual operating system's nucleus that is mapped guest virtual=guest real. This specification reduces the overhead associated with maintaining shadow page and segment tables VM/SP can selectively invalidate shadow table entries associated with pages stolen from within this V=R area.

The SET STBYPASS nnnnnK (or nnM) specification can be either approximate or precise. Generally, the approximate value is sufficient. However, address spaces may abend with address space errors. When these errors occur, then specify a precise value.

To determine an approximate nnnnnK (or nnM) value:

1. Issue the SET STBYPASS nnnnnK command with nnnnnK equal to the virtual machine storage size. This specification causes VM/SP to respond with the highest allowable high-water mark.
2. Reissue the SET STBYPASS nnnnnK command with nnnnnK equal to the highest allowable high-water mark from step 1.

Note: The highest allowable high-water mark may not be the true value. The virtual translation tables may, by chance, have several pageable page frames in the V=R storage area that map contiguously with the true high-water mark. If this is the case, address spaces may abend with address space errors. When these errors occur, use one of the following methods to determine the precise nnnnnK value.

To determine the precise nnnnnK (or nnM) value follow one of these methods:

- An installation can:

1. Locate entry CVTPVTP (X'164') in the SVS or MVS CVT. This is the address of the PVT.
2. Locate entry PVTFPFN (X'10') in the PVT. This is a half word value that represents the relative block number (RBN) of the first page frame table entry (PFTE) in the page frame table (PFT).
3. The value at PVTFPFN is left justified and the 12 high order bits are the 12 high order bits of a 24 bit address. Thus a value of X'0960' at PVTFPFN becomes X'096XXX' in an address. The 12 low order bits are zeroes, so the result is X'096000' for the address value you are looking for.

4. Take the X'096000' and convert it to decimal. The result is 614,400.
5. Divide the decimal value 614,400 by 1024. The result will be the nnnnnK value you are looking for, in this case 600K.

Notes:

1. For virtual operating systems that use only one address space (OS/VS1, DOS/VS, DOS/VSE), severe performance degradation can be avoided on the 168-3, 3032, and 3033 processors by issuing the STBYPASS command with a high-water mark equal to the size of the virtual machine. This forces CP to dispatch the virtual machine as if it were running with dynamic address translation (DAT) off no matter what the setting is in the virtual PSW. With DAT turned off, the virtual operating system is dispatched with 4K pages even though this operating system normally uses 2K page sizes. The use of 2K page virtual storage sizes results in a slower instruction rate because the high speed buffer is split in half and reset any time control register zero changes page sizes.
2. The nearer the value for nnnnnK (or nnM) to the virtual machine size, the greater the reduction in VM/SP overhead.
3. Also, the STBYPASS nnnnnK command should be used with single processor mode in AP and MP systems.

For more details about specifying the STBYPASS command, refer to VM/SP CP Command Reference for General Users.

To display the STBYPASS specifications, the user can issue the QUERY SET command.

Restrictions:

When using shadow table bypass for the V=V user, the following restrictions apply:

- Below the high-water mark, the virtual operating system must map each virtual address, starting from location zero, to its real address.
- When multiple segment tables are used by a virtual operating system, the page tables that correspond to the area below the high-water mark must be common to all segment tables.
- To invalidate entries below the high-water mark, the virtual operating system must specify DIAGNOSE code X'10' to release the virtual pages. However, the operating system is not restricted when validating entries below this mark.

STFIRST Directory Option

If virtual machine assist is available on the real machine, the STFIRST option must be in the virtual machine's directory to authorize the use of the SET STBYPASS command. Users can specify the STFIRST performance option in the OPTION directory control statement.

STFIRST should only be used for virtual machines that execute debugged and tested production workloads on these virtual operating systems: DOS/VS, OS/VS1, SVS, and MVS. STFIRST should not be specified for virtual machines executing test systems or programs that do not follow the programming restrictions for shadow table bypass. (These restrictions are listed in the topic "Shadow Table Maintenance Support" in this section of this publication.) Otherwise, either VM/SP could abnormally terminate or some other unpredictable result could occur.

SET STMULTI Command

When a virtual operating system uses more than one segment table, a user can issue the SET STMULTI command to define:

- How many shadow tables (maximum of 6) that CP is to support for the virtual machine -- n operand.
- The number of contiguous shadow page tables (in each pool of shadow page tables) for the virtual operating system's dynamic paging area -- USEG xx operand (SVS and MVS users only).
- The number of contiguous segments for the common area (at the high end of an address space) that is shared by all address spaces within the virtual operating system -- CSEG yyy operand (SVS and MVS users only).

Note: The purpose of the USEG and CSEG parameters is to improve CP performance by decreasing the number of shadow page tables that CP must maintain. Furthermore, USEG must be specified in order to specify CSEG. If these values are too low or the USEG parameter is specified without the CSEG parameter, poor virtual machine performance can result. Also, you can turn off the USEG or CSEG performance options by specifying a zero in either parameter. However, a zero value nullifies the performance benefits.

To display the STMULTI specifications, the user can issue the QUERY SET command.

Before using the STMULTI command, determine the values to specify on the command. The following descriptions explain a method for calculating these values.

The n Operand: Define this value according to the operating system used in the virtual machine:

- For SVS, specify two.
- For MVS, specify a value equal to the average number of initiators that are active at one time, plus two (to equal one address space for the nucleus and one address space for the master scheduler). (However, unless the value in field DMKSYSMS in module DMKSYS is also changed, the value cannot exceed the maximum of six.)

Because shadow table maintenance support limits the maximum number of STOBLOKs (segment table origin control blocks) to six, the actual number used depends upon the number of active MVS address spaces. If the demand for STOBLOKs exceeds the maximum number allowed, considerable STOBLOK stealing may take place and degrade the virtual machine's performance.

To monitor STOBLOK stealing, use the VM/SP Monitor. It collects these statistics about shadow tables in the class 4 code 1 record: (1) the maximum number allowed (the n operand), and (2) the actual number of active address spaces. To reduce these statistics and print them, use the VM/370 Performance/Monitor Analysis Program (VMAP), 5798-CPX.

The USEG xx Operand: Define this value based on the value of the page table steal counter in the ECBLOK (extension to VMBLOK for virtual machine with relocate).

By experimenting with different USEG values over various time periods, users can calculate an appropriate value for their operating systems. For example, when the increase in the counter value is high (a three- or four-digit hexadecimal value), increase the USEG value. When the increase in the counter value is low (a two-digit hexadecimal value), the USEG value is probably appropriate.

To automatically monitor the value of the page table steal counter, use the VM/SP Monitor. It periodically collects the value of this counter and other shadow table maintenance counters in the class 4 code 1 record. To reduce these statistics and print them, use the VM/370 Performance/Monitor Analysis Program (VMAP), 5798-CPX.

To manually locate the page table steal counter:

1. Enter (or have entered) this CP command (class C and E):

```
#cp loc userid
```

The userid in the LOCATE command is the name of the user's virtual machine. The command prints the address of this user's virtual machine block (VMBLOK).

2. Add X'0C' to the VMBLOK address to locate the pointer to the ECBLOK (VMECEXT field).
3. Locate the ECBLOK.
4. Add X'7C' to the ECBLOK address to locate the page table steal counter (EXTUPTST field).

The CSEG yyy Operand: Define this value to equal the number of 64K segments in the SVS or MVS common area.

To calculate this value:

1. Run the AMBLIST service aid program to find the beginning address of the PLPA.
2. Subtract the address found in step 1 from X'FFFFFF', and convert it from hexadecimal to decimal.
3. Divide the result from step 2 by 65,536 (64K) and round it to the nearest 64K segment.
4. Specify the decimal value in step 3 in the CSEG operand.

This calculation represents the maximum common area size. However, specifying this size for the CSEG operand may not provide the best performance.

For better performance, organize the SVS or MVS PLPA by packing frequently used modules together and putting them in the high address range of the PLPA. Then, subtract the size of this packed area from the maximum PLPA size. This CSEG value should represent the PLPA size from its beginning address to the lower address boundary of the packed area.

Note: For better performance in single processor mode, use a CSEG value that represents the maximum size of the common area.

To calculate this value:

1. Locate entry CVTSHRVM (X'1A0') in the SVS or MVS CVT.
2. Subtract the value in entry CVTSHRVM from X'FFFFFF' and convert the result to decimal.
3. Divide the result from step 2 by 65,536 (64K) and round it to the nearest 64K segment.
4. Specify the decimal value in step 3 in the CSEG operand.

Performance Guidelines

When run in a virtual machine, the performance characteristics of an operating system are difficult to predict. This unpredictability is a result of the complex interaction of many factors that affect performance. These factors can be broadly classified into three groups:

- Configuration factors
- Workload factors
- VM/SP performance factors

Performance of any virtual machine may be improved by the choice of hardware configuration, operating system workload, and VM/SP performance options. While a specific virtual machine's performance may not equal that of the same operating system running stand-alone on the same System/370, in some situations the total throughput obtained in the virtual machine environment can be equal to, or better than, that obtained on a real machine.

Configuration Factors

These hardware configuration factors influence the performance of an operating system in a virtual machine:

- The System/370 model used.
- The amount of real storage available.
- The speed, capacity, and number of paging devices.
- The degree of channel and control unit contention, as well as arm contention, affecting each paging device.

- Whether virtual machine assist or VM/370 extended control-program support is installed on the hardware and enabled.
- Interference between system paging devices and devices for processing a user's I/O requests.

When discussing these performance factors, this discussion assumes that the reader is familiar with the need to design an optimal configuration for a specific workload and operating system.

When moving a specific workload and operating system to the virtual machine environment, an installation should plan for an increased need in such hardware requirements as real storage, DASD space, and processor size. While VM/SP's overhead for dispatching, scheduling, and paging may be relatively small, its overhead for simulating privileged instructions may be considerable.

When not operating under VM/SP, an operating system runs directly on its own hardware (native mode) and manages its resources through the use of privileged instructions (such as SVC and LPSW) issued in supervisor state. When executing in a virtual machine, VM/SP dispatches the operating system in problem state, and any privileged instructions issued by the virtual machine cause a real privileged instruction exception interruption. This interruption transfers control to VM/SP to simulate the instruction. The amount of work done by VM/SP in analyzing and handling a virtual machine-initiated interruption depends upon the type and complexity of the interruption. Thus, any reduction in the number of privileged instructions issued by a virtual machine's operating system reduces the amount of extra work VM/SP must do to support that operating system.

Virtual machine assist support has been specifically designed to reduce VM/SP's overhead associated with simulating privileged instructions. It is the single, most effective method for reducing privileged instruction simulation. Any installation that is going to run a production operating system under VM/SP should consider virtual machine assist as a prerequisite for improving performance. Other steps for improving performance (such as using specialized VM/SP performance functions) are of secondary importance compared to using virtual machine assist.

VM/370 extended control-program support (ECPS: VM/370) is a hardware assist function that provides support over and above that provided by virtual machine assist support. It improves VM/SP performance beyond that attained by virtual machine assist support by reducing VM/SP's real supervisor state time needed to support virtual machines. VM/SP System Programmer's Guide lists the specific functions of ECPS: VM/370 that certain System/370 models support.

Workload Factors

These workload factors influence the performance of an operating system in a virtual machine:

- The type of operating system being used.
- The total number of virtual machines running under VM/SP.
- The type of work each virtual machine is doing, especially the amount of I/O processing required.

By measuring and evaluating the effect of these workload factors on a specific configuration, an installation can understand their effect on performance and know which steps to take to improve performance.

To relate these measurement values to system workload for a specific configuration, an installation must define its workload. The definition of workload varies with the environment:

<u>Environment</u>	<u>Workload Definition</u>
Interactive time-sharing system	Arrival rate of transactions and the processor time and working set size required for each transaction.
Batch system	Job throughput and resource requirements (processor time, region or partition size, and number of SIOs issued) for each job.

By using these workload definitions, an installation can measure its workload under VM/SP as follows:

<u>Environment</u>	<u>Workload Measurement</u>
CMS users under VM/SP	Number of active users
Operating system under VM/SP	User I/O requests executed

When both an operating system and CMS users run under the same VM/SP system, workload measurement depends upon which type of environment dominates the VM/SP system.

To measure workload performance in a specific configuration, an installation can use the Field Developed Program VM/370 Performance/Monitor Analysis program (5798-CPX). This program plots a number of important system variables (such as processor usage, various contention measurements, and paging rates) against workload measurement for both the CMS and operating system workloads under VM/SP. For a specific configuration, it allows an installation to relate processor usage, storage usage, and resource contention to the total system workload in both interactive and batch production environments.

By using this analysis program, an installation can eventually determine its optimum processor model, storage size, and I/O configuration for a specific workload. Thus, an installation may determine that it needs to do such things as: redistribute data sets to reduce arm contention, add control units and channels to reduce I/O contention, and add paging devices to reduce interference between system and user I/O processing.

VM/SP Performance Factors

These specialized VM/SP software techniques influence the performance of an operating system in a virtual machine:

- Whether VM/VS handshaking is used.
- The type and number of VM/SP performance options in use by one or more virtual machines.

VM/VS Handshaking: VM/VS handshaking (described earlier in this section under the topic "VM/VS Handshaking") permits instructions issued by an operating system in a virtual machine to be processed directly by the processor. It also permits VM/SP to simulate privileged instructions.

VM/SP Performance Options: After measuring the performance of both VM/SP and the virtual machines it supports, the system analyst and the general user can each use certain VM/SP performance options. These options allow them to create a special performance environment for one or more virtual machines. The options allow:

- The system analyst to redistribute system resources either to balance inequities or to favor one virtual machine over another.
- The general user to improve the performance for his virtual machine.

The VM/SP system operator, on behalf of the system analyst, can give certain options to a specific virtual machine to improve its performance over other virtual machines. A general user has certain performance options that give limited control over his virtual machine. The options available to the system operator and the general user are:

<u>System Operator</u>	<u>General User</u>
Locked pages option	Virtual machine assist
Reserved page frames	VM/370 extended control-program support
Priority	Virtual=real option ¹
Favored execution option	STBYPASS command

In the following list, the first option and either of the next two options can be applied to only one virtual machine at a time.

- Favored execution with guaranteed percentage
- Reserved page frames
- Virtual=real option¹

The following options can be applied to as many virtual machines as desired:

- Basic favored execution (without guaranteed percentage)
- Priority
- Virtual machine assist
- VM/370 extended control-program support (ECPS: VM/370)
- Locked pages

For basic information about these options, refer to the VM/SP System Programmer's Guide. For details about specifying the options for the system operator, refer to VM/SP Operator's Guide. For details about specifying the options for the general user, refer to VM/SP CP Command Reference for General Users.

¹This option cannot be specified in a command. To obtain it, a general user requests the VM/SP system programmer to specify it on the OPTION control statement (VIRT=REAL option) for the user's virtual machine directory entry.

The following performance-related additions to the VM/SP system control program are available. For certain environments, these additions:

- Improve throughput
- Provide better terminal response
- Reduce paging overhead by keeping the most active pages on preferred DASD, and migrating the inactive pages to slower devices
- Reduce overhead associated with maintaining shadow page and segment tables
- Improve performance for a production virtual storage operating system running under VM/SP
- Increase throughput of MVS running under VM/SP on an attached processor or multiprocessor system

PERFORMANCE MEASUREMENTS

Performance measurements apply to both the VM/SP system and the individual virtual machine. How well the system responds to the needs of the users is of prime importance to the general user. How efficiently the individual virtual machine makes use of the allotted storage, processor, and I/O facilities is of prime importance to the system analyst.

VM/SP provides certain CP commands (INDICATE and MONITOR) that allow both VM/SP and virtual machine performance to be tracked and measured; other commands allow the setting of certain options to improve performance. To reduce and help analyze the data produced by the MONITOR command, the Field Developed Program VM/370 Performance/Monitor Analysis program (5798-CPX) is available. By using this program, an installation can eventually determine its optimum processor model, storage size, and I/O configuration for a specific workload.

For a complete description of the INDICATE and MONITOR commands, refer to the VM/SP System Programmer's Guide.

EMPHASIZING INTERACTIVE RESPONSE TIMES

Most conditions for good performance, established for the time-sharing and batch systems, apply equally well to mixed mode systems. However, two major factors make any determination more difficult to make. First, get evidence to show that, in all circumstances, priority is given to maintaining good interactive response, and that nontrivial tasks really execute in the background. Second, background tasks (no matter how large, inefficient, or demanding) should not be allowed to dominate the overall use of the time-sharing system. In other words, in mixed mode operation, get evidence to show that users with poor characteristics are discriminated against for the sake of maintaining an efficient system for the remaining users.

A number of other conditions are more obvious and straightforward. For example: Measure response time and determine at what point it becomes unacceptable and why. Studies of time-sharing systems have shown that a user's work rate is closely correlated with the system response. When the system responds quickly, the user is alert, ready for the next interaction, and thought processes are uninterrupted. When the system response time is poor, the user loses concentration.

Generation Procedures Under VM/SP

VM/SP can help considerably throughout the system generation process. Probably VM/SP's biggest advantage is the ability to generate the system under VM/SP without disturbing the normal production activity.

The system programmer (or whoever is responsible for the operating system) can logon to his own virtual machine and go through the generation steps at his own pace while the daily work is being processed. He can use the VM/SP CMS editor to create and update the job streams that are used during system generation. Whenever the system generation process requires, he can use CMS EXEC procedures to pass these saved job streams to the test system. When the system is tested, it can be placed online, replacing the previous version with minimal interruption to the production activity.

For a discussion of the CMS editor and EXEC facilities, refer to the VM/SP CMS User's Guide. For details about the system generation procedure for DOS/VS and OS/VS under VM/SP, refer to these system-dependent sections in this publication.

Creating VM/SP Directory Entries

To allow a virtual machine to exist in the VM/SP system, the VM/SP system requires a directory entry definition. Each definition is kept in a directory entry source file (filetype DIRECT) on a user minidisk. An installation must use the VM/SP directory program to convert these source definitions in the VM/SP system directory file (usually on the system residence disk) that contains one entry for each virtual machine.

Each directory entry contains a number of directory control statements that define the virtual machine's configuration and other operational characteristics to VM/SP. In general, a virtual machine configuration defined in the directory consists of the following:

- Virtual storage, console, and processor
- Direct access storage devices
- Unit record devices
- Other devices

Figure 12 shows the relationship of a directory entry to both the VM/SP system's real devices and the virtual machine's virtual devices. The installation must keep both the source and system directories updated. As users submit additions and/or changes, the installation must either create new or update current directory entries. This updating can be done by using the VM/370 Directory Maintenance Program Product (5748-XE4), the CMS editor, or punched cards. (For more details about this program product, refer to the VM/370 Directory Maintenance Program Product General Information Manual, GC20-1836.)

To create directory entries for operating systems running in virtual machines, installations must consider both the general and unique requirements for specifying directory entries. For general details about specifying directory entries, refer to VM/SP Planning and System Generation Guide. For unique details about specifying directory entries for operating systems running in a virtual machine, refer to the following topic "Unique Directory Entry Considerations."

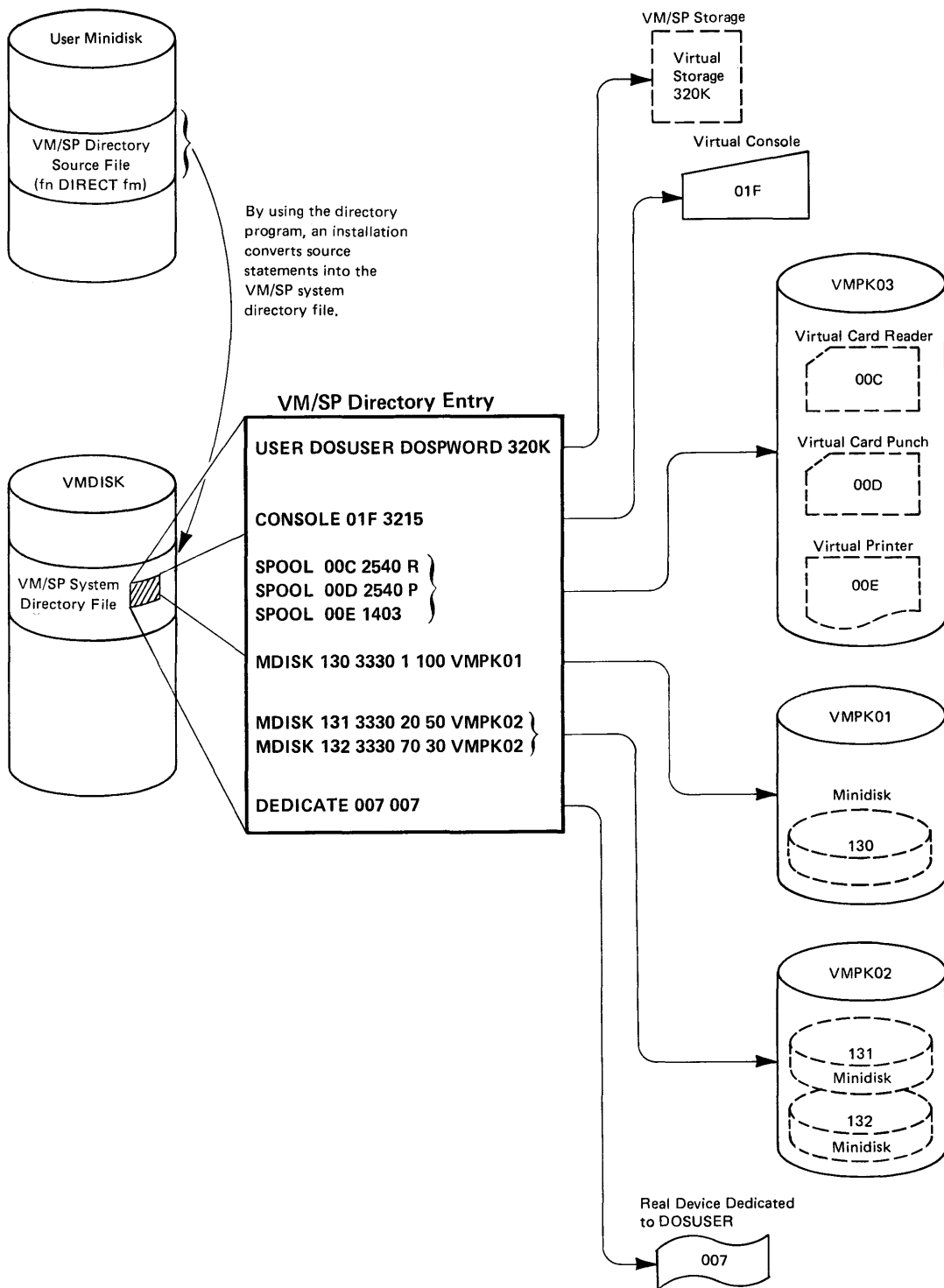


Figure 12. Relationship of VM/SP Directory Entries

UNIQUE DIRECTORY ENTRY CONSIDERATIONS

This topic lists directory control statements in their logical order of appearance in a directory entry. It describes only those statements with unique considerations for running an operating system in a virtual machine.

USER Control Statement

The USER control statement has no unique considerations for running an operating system in a virtual machine.

ACCOUNT Control Statement

The ACCOUNT control statement has no unique considerations for running an operating system in a virtual machine.

OPTION Control Statement

The OPTION control statement allows virtual machine users to specify certain options and features for their virtual machines.

ACCT Option

The ACCT option allows one user to charge another user for virtual machine resources. For example: If the machine is performing a batch type operation, a virtual machine user can generate job accounting records for each job processed. To generate accounting records for a virtual user, refer to the VM/SP System Programmer's Guide for a discussion of DIAGNOSE code X'4C'.

ECMODE Option

Specify the ECMODE option if the virtual machine uses an operating system that:

- Runs in extended control mode
- Uses dynamic address translation (DAT)
- Uses extended control registers other than zero
- Addresses I/O channels 6 through 15

If this option is not specified in the directory, a user can enter EC mode by issuing the CP SET command with the ECMODE operand:

```
#cp set ecmode on
```

When the ECMODE option is specified for a virtual machine, the saved segments of the virtual operating system are shareable.

ISAM Option

The ISAM option allows VM/SP to handle self-modifying channel programs that ISAM generates for some of its operations. If this option is not specified in the directory entry, users can obtain the ISAM facility by issuing the CP SET command with the ISAM operand:

```
#cp set isam on
```

The ISAM option must be specified if a user is:

- Using ISAM in an OS/PCP, OS/MFT, or OS/MVT virtual machine
- Using ISAM in a V=R partition or region of an OS/VS virtual machine
- Using VS1 handshaking with VS1 nonpaging

Do not specify the ISAM option if a user is:

- Using ISAM in a DOS or DOS/VS virtual machine
- Using ISAM in a V=V region of an OS/VS virtual machine

REALTIMER Option

Enter the REALTIMER option if the virtual timer is to be updated during virtual wait time as well as during virtual processor time. This option is required for operating systems running applications where certain interruptions are timer driven. If this option is not specified in the directory entry, a user can obtain this timing facility by issuing the CP SET command with the TIMER operand:

```
#cp set timer real
```

To turn off the option, issue:

```
#cp set timer off
```

STFIRST Option

When virtual machine assist is available on the machine, the STFIRST option permits the virtual machine user to issue the SET STBYPASS nnnnnK command.

The SET STBYPASS nnnnnK command defines the high-water mark (or highest limit) of the virtual operating system's nucleus that is mapped guest virtual=guest real. This specification allows VM/SP to reduce the overhead associated with maintaining shadow page and segment tables under CP. Thus, VM/SP can now selectively invalidate shadow table entries associated with pages stolen from within this guest V=R area, rather than invalidating the entire table.

The STFIRST option should be used for virtual machines that execute production workloads on these virtual operating systems: DOS/VS, OS/VS1, SVS, and MVS. Users should not specify STFIRST for virtual machines executing test systems or programs that do not follow the programming restrictions for shadow table bypass. (These restrictions are listed in the topic "Shadow Table Maintenance Support" in this section of this publication.) Otherwise, either VM/SP could abnormally terminate or some other unpredictable result could occur.

SVCOFF Option

The SVCOFF option specifies that VM/SP, rather than virtual machine assist or ECPS: VM/370, handle all SVC interruptions. To override this option, issue the CP SET command:

```
#cp set assist svc
```

Note: If the operating system uses SVC 76 for error recording, VM/SP handles the SVC 76 interruptions whether SVCOFF is in effect or not.

Virtual=Real Option

The virtual=real option may be desirable, undesirable, or mandatory. It is desirable when running a virtual machine operating system (like DOS/VS or OS/VS) that performs its own paging because it eliminates the possibility of double paging. The virtual=real option is not desirable when running an operating system in nonpaging mode with VM/VS handshaking in a virtual machine. It is mandatory to use the virtual=real option to allow programs that execute self-modifying channel programs or have a certain degree of hardware timing dependencies to run under VM/SP.

The virtual=real area is set up at VM/SP IPL. The primary VM/SP system operator can release the area for use as part of the dynamic paging area. Once released, it cannot be reclaimed except by reloading VM/SP. The virtual=real area must be released in total; that is, unused pages of the area cannot be selected for release.

There are several ways to use the virtual=real option effectively on a data communication system with no CCW translation (SET NOTRANS ON) that has multiple ports. Dedicate either the transmission control unit or communications lines to that system via the ATTACH command or by VM/SP directory assignment. Conversely, on a multiple port data communication virtual=real operation, virtual 270x lines (that is, lines assigned and used by the CP DEFINE and DIAL commands) operate with CCW translation. When VM/SP detects the use of nondedicated communication lines, it ignores the SET NOTRANS ON command.

Note: By issuing the SET STBYPASS VR command, a user eliminates shadow tables and the overhead associated with maintaining them. VM/SP modifies the virtual operating system's page table to relocate virtual page zero to the highest real address within the V=R area. It is then possible to dispatch the virtual machine and have VM/SP's real control register 1 point to the virtual machine's page and segment tables. To terminate the shadow table bypass function, issue the SET STBYPASS OFF command. For details about how to specify this command, refer to VM/SP CP Command Reference for General Users.

For general information on specifying a virtual=real machine and on defining the VIRT=REAL option in a virtual machine's directory entry, refer to VM/SP Planning and System Generation Guide.

Note: Portions of the DOS/VS supervisor, OS/VS1 or OS/VS2 nucleus may have to be relocated above page zero to keep input operations from compromising VM/SP's page zero. For a more detailed description of the problem and the suggested solutions for DOS/VS, OS/VS1, and OS/VS2, refer to the "System Generation Recommendations" topic in the sections for DOS/VS and OS/VS in a virtual machine in this publication.

By specifying the VIRT=REAL option, the virtual machine is eligible to occupy VM/SP's low storage. Thus, with the exception of page 0, all other virtual storage addresses correspond to the real storage addresses. VM/SP's page 0 occupies the first 4096 bytes of real storage, and VM/SP relocates virtual page 0 to a position immediately following the area set aside for V=R operation (see Figure 13).

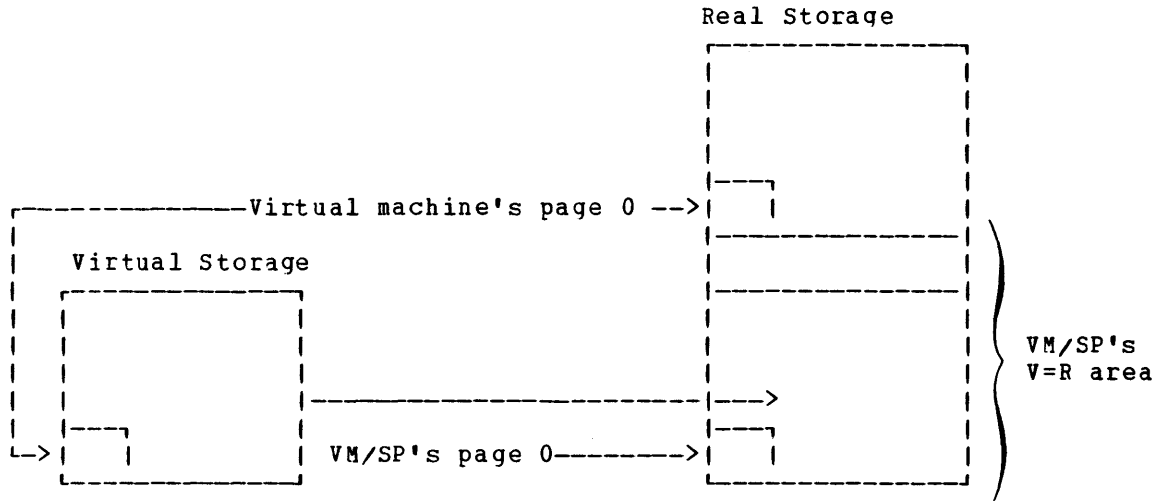


Figure 13. Virtual=Real Machine

This option can be specified for many virtual machines; however, only one virtual machine can occupy the V=R area at one time. If this option is specified and the V=R area is occupied when logging on, VM/SP creates the virtual machine in virtual=virtual mode and sends a message to inform the user of this development.

VMSAVE Option

If a virtual machine user specifies the VMSAVE option (in conjunction with predefined values in the NAMESYS system generation macro instruction), VM/SP automatically saves the virtual machine's contents. This is done either when VM/SP terminates the virtual machine or when VM/SP itself abnormally terminates. After logon, a user can restore the contents by issuing the IPL command. In this command, the user specifies the name of the save area that preserves the contents of the virtual machine. While the VMSAVE option can be used by any virtual machine, it is most useful for production workloads on data base/data communication systems such as IMS.

To enable the VMSAVE option for virtual machines, either use the VMSAVE directory option or issue the SET VMSAVE area-name command. For details about how to use the VMSAVE option, refer to VM/SP System Programmer's Guide. For details about how to use the SET VMSAVE area-name command, refer to VM/SP CP Command Reference for General Users.

370E Option

The 370E option lets a user specify whether a virtual machine can use the MVS/System Extensions functions of the VM/SP Program Product¹. Enable the MVS/System Extensions functions by:

- Specifying the 370E option in the OPTION control statement of the virtual machine's directory.

--or--

- Issuing the class G command SET 370E ON provided the system operator has issued the class A command SET S370E ON.

To disable these functions for the VM/SP system, system operators can issue the class A command SET S370E OFF. To disable these functions for a specific virtual machine, general users can issue the class G command SET 370E OFF.

To display the ON and OFF system status of the MVS/System Extensions functions, both classes of users can issue the QUERY S370E command. To display the ON and OFF status of these functions for a specific virtual machine, general users can issue the class G command QUERY SET.

Note: An RPQ (MK3272) is available for the 158-3 processor that allows coexistence of virtual machine assist and the S/370 Extended Facility (S370E) and S/370 Extended Feature. Thus, an MVS/SE virtual machine can run under VM/SP with virtual machine assist active on a 158-3 processor.

IPL Control Statement

If a virtual machine runs one operating system most of the time, a user can have that system automatically loaded each time he logs on. Use the IPL statement to identify the operating system to VM/SP, such as by specifying the following:

```
ipl 350
```

The virtual address 350 represents the address of the device that contains the system to be loaded. Or, if the virtual machine's operating system has been "saved" (by using the CP SAVESYS command), specify:

```
ipl DOSVS
```

Where DOSVS is the name under which the system was saved.

Note: For the VM/SP system operator to automatically logon to a virtual machine (by using the class A or B AUTOLOG command), the virtual machine directory entry must contain an IPL control statement.

¹This program product supports the System/370 Extended Facility and the System/370 Extended Feature. For a list of the appropriate processors supported by the Facility and the Feature, refer to VM/SP Planning and System Generation Guide.

CONSOLE Control Statement

The CONSOLE control statement has no unique considerations for running an operating system in a virtual machine. However, a secondary userid must be specified in this directory statement whenever you want to utilize the single console image facility. The secondary userid denotes another user that controls all messages, replies, and commands for a virtual machine after the primary user disconnects. This gives an operator added flexibility in an environment where service virtual machines are used because the operator can control several disconnected machines (via CP SEND command) from one physical terminal.

MDISK Control Statement

The MDISK control statement has no unique considerations for running an operating system in a virtual machine.

SPOOL Control Statement

The SPOOL control statement has no unique considerations for running an operating system in a virtual machine. The SPOOL control statement supports the specification of a virtual 3800 printer for use by the virtual machine.

DEDICATE Control Statement

Use the DEDICATE control statement to provide a virtual machine with a corresponding real device. The virtual machine has sole use of the dedicated device.

Magnetic Tapes: A device such as a magnetic tape drive can be used by only one virtual machine at a time; therefore, specify it in a directory entry with a DEDICATE statement.

Example: The directory specifies:

```
DEDICATE 181 281
```

This statement allows the operating system to access the device at real address 281 via a virtual address of 181.

Unit Record Devices: In most cases, spooling represents the most efficient way of handling the unit record input and output of many virtual machines. However, special cases may justify the dedication of a real unit record device to a single virtual machine.

One special case is when the virtual machine's operating system does its own spooling, such as POWER/VS under DOS/VS or JES under OS/VS. To eliminate double spooling of printer output, include a DEDICATE statement in the virtual machine's directory entry, such as:

```
DEDICATE 00E 002
```

This statement causes VM/SP to pass all virtual printer 00E output directly to the real printer at 002.

Another case where a user may want a unit record device dedicated to his virtual machine would be if the virtual machine produced a sufficient volume of output to keep the device busy.

Users can also have the system operator dynamically dedicate a unit record device to their virtual machine. First, send the system operator the message:

```
#cp msg operator pls attach punch at 00d
```

If a punch is free at 00d, the operator issues the command:

```
attach 00d to userid as 00d
```

When the device is attached, VM/SP sends a confirmation message to userid:

```
PUN 00D ATTACHED
```

When the device is no longer needed, issue the command:

```
#cp detach 00d
```

Remote Devices: The DEDICATE statement can be used to attach remote 3270 Information Display System Printers (3284, 3286, 3287, 3288, and 3289) to a virtual machine.

For example: A directory entry can include the statement:

```
DEDICATE NETWORK 00E 1002
```

where 00E is the virtual address of the device in the virtual machine and 1002 is the resource ID as specified in DMKRIO. Remote 3270 Information Display System Printers can also be attached by the NETWORK ATTACH command. For more details, see the VM/SP Operator's Guide.

Unsupported Devices: The DEDICATE statement can be used to place a device that VM/SP does not support into a virtual machine's configuration. To dedicate a device, the device must:

- Be physically connected to the System/370
- Be supported by the virtual machine's operating system
- Not violate any of the restrictions contained in the VM/SP restriction section of the VM/SP Planning and System Generation Guide

For example: A directory entry can include the statement:

```
DEDICATE 007 012
```

where real address 012 could represent a 2671 Paper Tape Reader that is part of the System/370 on which VM/SP is running. If the operating system was generated with a 2671 defined at address 007, VM/SP handles the device and CCW address translation associated with reading from the device. The operating system in the virtual machine is responsible for error recovery and error recording procedures.

2305 Devices: When using the DEDICATE statement to attach a 2305 to a virtual machine, both the real and virtual addresses must refer to the first base device address on the unit. The first base address of the 2305 is 0 or 8 -- the resulting address appears as xx0 or xx8. However, when VM/SP processes the statement, it creates all eight addresses (0-7 or 8-F) for the 2305.

LINK Control Statement

The LINK control statement has no unique considerations for running an operating system in a virtual machine.

SPECIAL Control Statement

Use the SPECIAL control statement to add I/O devices that do not require corresponding real devices. Some examples are: magnetic tapes, channel-to-channel adapters, pseudo timers, communication lines, and devices that, while available to the System/370, are not supported by VM/SP.

Installations can use the SPECIAL control statement to specify a virtual transmission control unit for a multiple-access operating system (MVS/TSO, CICS). For example: If the system requires three communication lines from a 2703, specify:

```
SPECIAL 061 2703 IBM
SPECIAL 062 2703 IBM
SPECIAL 063 2703 TELE
```

Before a terminal can communicate with a multiple-access system, the terminal user must issue the DIAL command:

```
dial userid
```

to connect to any available line port; or issue:

```
dial userid 062
```

to connect to a particular line.

Note: Of the three lines specified in the foregoing example, one was a teletypewriter line and two were IBM terminal lines. When the DIAL command is issued with no specific address, VM/SP connects the terminal to any available line as defined in the SPECIAL control statement; the line then belongs to the specified userid. If no lines are available or if all lines are busy, VM/SP issues an error message and does not make the connection. To drop a dialed line, a user must issue the CP RESET command.

DEFINING VIRTUAL DEVICES

When using the SPOOL, DEDICATE, and SPECIAL control statements to define virtual devices, the rules for assigning virtual addresses are the same as for real devices and control units. The type of subchannel required by a device's control unit dictates the valid address assignments. Devices that need special I/O interface protocol from control units, such as shared subchannels, require that all 16 device addresses (0-F) be reserved. Therefore, you can only attach similar devices to a control unit with a shared subchannel. For devices that need a control unit with a nonshared subchannel, only one address per device is required. Devices that attach to a control unit with a nonshared subchannel do not have to be the same type.

For example: If the directory entry specifies:

```
SFOOL 102 3211
SPECIAL 103 3270
```

the 3270 specified at address 103 requires a shared subchannel and therefore reserves addresses 100-10F for display type devices. Since the device specified at address 102 is not a display unit but a printer, processing of channel programs involving these two devices can result in a hung or busy condition.

AUTOLOG FACILITY

AUTOLOG is a convenient way to initiate large production operating systems with many I/O devices that run under VM/SP. The I/O devices needed by these operating systems require considerable contiguous storage space for the I/O control blocks established by VM/SP. When these large operating systems are started after other, smaller users have been using VM/SP, the contiguous storage space may not be available. When there is insufficient contiguous space, the logon of the virtual machine is successful; however, there may be an insufficient number of I/O devices to run the operating system and its application programs.

To ensure sufficient contiguous storage space, logon those virtual machines after loading VM/SP.

- Have the VM/SP system operator issue the CP AUTOLOG command before enabling user terminals.
- Define the AUTOLOG1 virtual machine in the VM/SP directory. The AUTOLOG1 userid can be used to logon and load virtual machines that require substantial contiguous storage.

Using the CP AUTOLOG Command

Before enabling user terminals, the VM/SP system operator can issue the CP AUTOLOG command for each production virtual machine that requires substantial contiguous storage. The directory entry for the userid indicated by the CP AUTOLOG command must contain an IPL statement for the desired operating system. For more information about the CP AUTOLOG command, refer to the VM/SP Operator's Guide.

Defining AUTOLOG1 in the Directory

After the VM/SP console operator loads the VM/SP system, VM/SP initiates one predefined operating system to run as a virtual machine without operator intervention, provided these two conditions are met:

1. An AUTOLOG1 virtual machine is defined in the directory.
2. The AUTOLOG1 directory definition includes an IPL control statement specifying an operating system.

Note: The AUTOLOG1 facility initiates a virtual machine with an operating system in disconnect mode. If that virtual machine issues a READ command to its own unavailable console, VM/SP terminates the virtual machine. WRITE commands to an unavailable operator's console merely cause the loss of information that is not spooled; no termination occurs. However, if a secondary user was specified for this virtual machine on the CONSOLE directory control statement, READs and WRITEs to the console are possible through the console of the secondary user.

To use AUTOLOG1 to initiate several virtual machines, have the VM/SP directory statements load CMS for the AUTOLOG1 userid. The CMS PROFILE EXEC would contain several CP AUTOLOG commands. Each AUTOLOG command initiates one virtual machine containing a production operating system. When using the CP AUTOLOG command, the directory entries referenced by the CP AUTOLOG command must contain an IPL statement.

When the production virtual machine is loaded directly by an IPL statement in AUTOLOG1, an operating system user gains access to the virtual machine in one of two ways:

1. By logging on as AUTOLOG1 with the appropriate password, or
2. Through the secondary user's console if a secondary user was specified on the CONSOLE directory control statement

When the production virtual machine is loaded (as a result of a CP AUTOLOG command in the CMS PROFILE EXEC), an operating system user gains access to the virtual machine by logging on with the userid specified in the CP AUTOLOG command or through the secondary user's console.

When the user logs off, contiguous storage space is relinquished. If the user wants to keep the virtual machine's I/O blocks in contiguous storage and temporarily relinquish use of the virtual machine, the user issues the CP DISCONN command. To reestablish usage, the user issues the CP LOGON command to reconnect to the virtual machine.

Single System With AUTOLOG1

The following example uses the AUTOLOG1 facility with an IPL statement.

```
USER AUTOLOG1 PASSWORD 256K
  ACCOUNT ACCTNO BIN3
  IPL 350
  OPTION ACCT
    CONSOLE 01F 3215 SECUSER
    SPOOL 00B 2501
    SPOOL 00C 2540 R
    SPOOL 00D 2540 P
    SPOOL 00E 1403
    MDISK 350 3330 101 30 OSDOS1 W
    MDISK 351 3330 1 20 UDISK1 W
```

With these directory entries, the operating system would be logged onto the VM/SP system in disconnect mode. A user accesses the system through the secondary user's console if a secondary user was specified on the CONSOLE directory control statement, or by logging on as userid AUTOLOG1 with the appropriate password. To temporarily relinquish use of the system without relinquishing contiguous storage, the user issues the CP DISCONN command. To reestablish use, a user issues the CP LOGON command. Issuing the CP LOGOFF command releases the contiguous storage space containing the VM/SP I/O device control blocks supporting the operating system.

Multiple Systems With AUTOLOG1

In this example, AUTOLOG1 initializes CMS in a virtual machine. The virtual machines containing the production operating systems are automatically logged on in disconnect mode from the CMS PROFILE EXEC. The CMS PROFILE EXEC contains several CP AUTOLOG commands, one for each virtual machine to be loaded. For each userid identified in a CP AUTOLOG command, there must be an IPL statement in the VM/SP directory to load the appropriate operating system into the virtual machine. The last CP command in the PROFILE EXEC may logoff AUTOLOG1. The virtual machines are logged onto VM/SP in disconnect mode.

AUTOLOG1 Virtual Machine

```
USER AUTOLOG1 PASSWORD 256K 1M AB
ACCOUNT ACCTNO BIN1
IPL CMS
    CONSOLE 009 3215
    SPOOL 00C 2540 R
    SPOOL 00D 2540 P
    SPOOL 00E 1403
    LINK VMSYS 190 190 RR
    MDISK 191 3330 1 10 UDISKA WR RPASS WPASS
```

PROFILE EXEC

```
&CCNTROL OFF NOMSG NOTIME PACK
CP SPOOL CONSOLE START
SET RDYMSG SMSG
CP SPOOL PRINTER CLASS A
CP SET EMSG TEXT
CP SET LINEDIT ON
CP AUTOLOG DOSUSER PASSWORD
CP AUTOLOG DOSVUSER PASSWORD
CP AUTOLOG OSUSER PASSWORD
CP LOGOUT
&EXIT
```

By having the preceding AUTOLOG1 directory entry and PROFILE EXEC, the DOSUSER, DOSVUSER, and OSUSER virtual machines (specified in the PROFILE EXEC) are now logged onto the VM/SP system in disconnect mode. A user accesses these virtual machines through their secondary user's consoles, if any, or by logging on with the userid of DOSUSER, DOSVUSER, or OSUSER along with the appropriate password. To temporarily relinquish use of one of these virtual machines without relinquishing contiguous storage, a user issues the CP DISCONN command. To reestablish use, a user issues the CP LOGON command. Issuing the CP LOGOFF command releases the contiguous storage space containing the VM/SP virtual I/O device control blocks.

SAMPLE DIRECTORY ENTRIES

This topic shows some useful virtual machines that can be defined when running operating systems in virtual machines. Sample directory entries for running specific operating systems under VM/SP are in the operating system dependent sections in this publication.

A Multiple-Access Virtual Machine

The following directory entry represents a multiple-access TSO system configured to handle one to four concurrent remote terminals and one local 3270. It has been given the VIRT=REAL option to improve response time.

```
USER TSOSYS PASSWORD 384K
ACCOUNT ACCTNO BIN8
IPL 290
OPTION REALTIMER VIRT=REAL ECMODE
    CONSOLE 01F 3215
    SPOOL 00C 2540 R
    SPOOL 00D 2540 P
    SPOOL 00E 1403
    DEDICATE 290 TSOSYS
    DEDICATE 291 ISOWRK
    SPECIAL 070 3270
    SPECIAL 080 2702 IBM
    SPECIAL 081 2702 IBM
    SPECIAL 082 2702 IBM
    SPECIAL 083 2702 TELE
```

A VM/SP Virtual Machine (TESTSYS)

The directory entry for this virtual machine has the ECMODE and REALTIMER options, allowing its user to check a new CP nucleus before moving that nucleus to the production system. TESTSYS contains two minidisks, 350 and 351. These disks are exact copies of the real system residence and scratch volumes. They are formatted and allocated so that the real system can spool and page on these disks. If a user needs additional disks, link to them before loading the virtual VM/SP system.

```
USER TESTSYS PASSWORD 512K
ACCOUNT ACCTNO BIN11
OPTION ECMODE REALTIMER
    CONSOLE 01F 3215
    SPOOL C 2540 R
    SPOOL D 2540 P
    SPOOL E 1403
    LINK VMSYS 190 190 RR
    MDISK 191 3330 16 10 UDISKA WR RPASS WPASS
    MDISK 350 3330 1 15 SYSWRK WR RPASS WPASS
    MDISK 351 3330 16 20 SYSWRK WR RPASS WPASS
```

Summary

To run any operating system in a virtual machine, an installation should:

- Design new and existing application programs to operate efficiently in a paging environment; that is, have them use VM/SP paging instead of DOS/VS or OS/VS paging.
- Reduce a virtual machine's I/O operations.
- Use VM/SP services for performance and communication, such as the VM/SP virtual machine options and VMCF.

When running specific multiprogramming operating systems under VM/SP (such as DOS/VS or OS/VS), an installation should consider how that system interacts with VM/SP -- especially when that system has a page wait or I/O wait. To interact with these systems, VM/SP provides VM/VS handshaking for certain DOS/VS and VS1 systems, and the diagnose interface for virtual operating systems, such as DOS/VS and OS/VS.

Other areas to consider when running multiprogramming operating systems under VM/SP are spooling, channel model-dependencies, whether to use multiple or alternate consoles, and the states or conditions of virtual devices (dedicated, shared, and spooled).

VM/SP also supports alternate paths, multiple-access virtual machines, operating systems using DASD reserve/release, and the ASP virtual machine. Installations can also alternate between operating systems under VM/SP.

While difficult to predict, performance of any virtual machine may be improved by the choice of hardware, operating system, and VM/SP options. VM/SP also provides the INDICATE and MONITOR commands to track and measure both VM/SP and virtual machine performance.

VM/SP can help considerably throughout the system generation process. Its biggest advantage is allowing an installation to generate a system under VM/SP without disturbing production activity.

To allow virtual machines to access the VM/SP system, the VM/SP system requires a file of directory entries that contains one entry for each virtual machine. Each directory entry contains a number of directory control statements that define the virtual machine's configuration and operational characteristics to VM/SP. Some directory statements have unique considerations when running an operating system in a virtual machine. There is an AUTOLOG facility to automatically initiate large production operating systems with many I/O devices under VM/SP.

Section 2. VM/SP in a Virtual Machine

This section describes how to update and test a VM/SP system in a virtual machine. After testing, use the DASD dump restore (DDR) program to dump the virtual CP (VM/SP) system to tape. Then, restore that virtual CP system to the real system residence disk. After performing these steps, an installation is ready to execute the new version of VM/SP with a minimum amount of real processor time.

VM/SP Directory Definition

Before an installation can test VM/SP in a virtual machine, it must first have a VM/SP directory entry for a VM/SP virtual machine. The virtual directory need only specify a minimum number of users that are sufficient to perform the testing. It is usually beneficial to define an operator's virtual machine that is large enough and varied enough to perform all necessary functions. This specification allows most virtual testing to be done from one userid. It does not require several userids to dial into this system to accomplish a test.

In the following sample directory entry, assume that TESTSYS is the userid for this virtual machine. TESTSYS has the options and configuration necessary to define a minimum system for initializing VM/SP in a virtual machine. A sample VM/SP directory entry for TESTSYS would be:

```
USER TESTSYS PASSWORD 512K
  ACCOUNT NUMBER BIN11
  OPTION ECMODE REALTIMER
  CONSOLE 01F 3215
  SPOOL C 2540 READER
  SPOOL D 2540 PUNCH
  SPOOL E 1403
  LINK CMSSYS 190 190 R
  MDISK 330 3330 1 15 SYSWRK WR RPASS WPASS
  MDISK 331 3330 16 20 SYSWRK WR RPASS WPASS
```

where:

The USER statement defines the userid as TESTSYS, the password as PASSWORD, and 512K storage (the minimum amount of storage to load a virtual VM/SP system).

The OPTION statement specifies the ECMODE and REALTIMER options, which are required for the virtual machine to operate in extended control mode and to wait for a timer interruption to continue processing.

The CONSOLE and SPOOL statements specify the console and spool device addresses. These addresses must match the same addresses as the real machine configuration. If that configuration is not used for the virtual system operation, they must match whatever configuration is specified in the DMKRIO module.

The LINK statement specifies that this virtual machine may operate CMS, although special considerations described later in this section have to be used to operate CMS.

The MDISK statements for 330 and 331 define disks for the CP system residence, paging, and spooling volumes.

Note: This directory entry configuration does not define any other user disks, teleprocessing lines, or tape drives. All additional devices required for testing VM/SP in a virtual machine can be specified by using the ATTACH, LINK, and DEFINE commands.

Virtual Machine Configuration

To run the VM/SP nucleus in a virtual machine, load it onto the minidisk that represents the virtual system residence volume. Then, before initializing the system, verify that the virtual machine configuration has:

- The correct console address
- Sufficient unit record devices available at the correct addresses
- Enough disks (either linked or attached) to make a reasonable test

When setting up the virtual machine configuration, a user can link to other user disks so that the VM/SP system can use these disks in its virtual operation. However, the user must ensure that links to other disks use the same addresses and device types as were specified in the DMKRIO module of the quest VM/SP.

For example: A real VM/SP system has 2314s defined as 130 to 137 and has 3330s defined as 330 to 337. To avoid operational errors, the virtual VM/SP system links to user 2314 disks in the range of 130 to 137 and links to user 3330 disks in the range of 330 to 337. If a user disk is linked to as a 2314 address when it is actually a 3330 or 3340 device, the virtual VM/SP system receives errors when trying to process that user disk.

DEFINING A CONSOLE FOR VM/SP IN A VIRTUAL MACHINE

Since the logon console for a virtual machine operates as a 3215, 3210, 3270, or 1052, one of the following two methods can be used to satisfy the console requirements for your VM/SP virtual machine:

1. In the DMKRIO for the second level VM/SP system you are building, define the console device as DEVTYPE 3215, 3210, or 1052 in the RDEVICE macro (Ex: RDEVICE ADDRESS=01F, DEVTYPE=3215).
2. Another approach is attaching a console-type device to your virtual machine and using that as your second level VM/SP console. For example, if your DMKRIO for address 01F defines a DEVTYPE of 3277, then attach a real 3277 to your virtual machine as address 01F to function as your second level VM/SP console.

Note: Regardless of which method you choose, you must specify 3215 in the CCNSOLE directory statement when defining your virtual VM/SP operator.

CMS SYSTEM

For VM/SP in a virtual machine to also run CMS, it must have access to the CMS system residence volume. The virtual machine can access this volume either before logon (by using the LINK statement in the directory entry) or after logon (by using the CP LINK command). If passwords are provided, the virtual VM/SP system can link to other users' disks so that they can be used as primary disks by the CMS system. Thus, under the one userid for the virtual VM/SP system, this system can access all the disks necessary to do a virtual system VMFLOAD or any other similar function.

2305 DEVICES

It is probably not necessary to have a 2305 paging device in the configuration unless the test specifically requires that device. If this device is required and if the real configuration allows it, the user may define temporary 2305 space for a paging volume.

ACCESSING A VIRTUAL VM/SP SYSTEM

Depending upon the nature of virtual machine testing, one or more teleprocessing lines can be defined so that users may use the DIAL command to access the VM/SP system in a virtual machine. In most cases, simple tests do not require teleprocessing lines to be defined or enabled at the virtual machine level. Most testing can be performed by the operator's virtual machine from the virtual console.

CP DISKS FOR THE VIRTUAL MACHINE

Before VM/SP in a virtual machine can use the CP disks for the virtual system residence, paging, and spooling volumes, an installation must first format and allocate space for these disks.

Formatting the Volumes

To format the system residence, paging, and spooling volumes, use the CP format/allocate program. Although this program can run in a virtual machine, it cannot run under CMS. To run the format program, make it available to the virtual machine by reading it into the virtual machine's spool card reader. This can be accomplished by spooling the program from a real card reader or from a CMS file in another virtual machine to userid TESTSYS in this example. Then, IPL it from that reader (IPL 00C). Because a virtual disk is being formatted, the cylinder or block specification should reflect the size of the virtual disk being used. For example: In the sample directory entry for TESTSYS (described earlier in this section), the MDISK statement for the virtual disk at real address 330 defines only 15 cylinders for the device; thus, only 15 cylinders on the virtual disk at real address 330 can be formatted.

When the virtual machine is going to use the same DMKSYS module that the installation is using, the virtual disk label should match the label in the installation-owned list. Thus, if an installation has two volumes in the owned list (such as CPDSK1 and CPDSK3), then those volume labels must match the minidisk labels used by the virtual machine.

Allocating Space for the Volumes

After formatting the volumes, allocate space on them to hold:

- A virtual directory
- Nucleus area
- Warm start area
- Error recording area
- Paging space
- Spooling space
- Dump space

If the space is inaccessible to the virtual VM/SP system (that is, beyond the size of the virtual disk), it must be assigned as permanent space. In this example, cylinders 16-99 on virtual device 330. This is necessary because the minidisk is smaller than the real device and by allocating it as permanent space, VM/SP will not access the area outside the minidisk. Otherwise, the virtual system attempts to access temporary space beyond the size of the virtual disk, resulting in the real VM/SP system reflecting either seek checks or command rejects to the virtual system.

When allocating permanent space, organize the cylinders to hold the directory, CP nucleus, error recording area, and the warm start area. Also, organize the areas to begin with the first cylinder or block available on the disk. If the real system residence volume uses this same organization, then the disk you use for your virtual system residence volume can use the same DMKSYS and DMKRIO modules.

For example: If the real configuration specifies permanent space, then the installation must generate a special DMKSYS module for the VM/SP system in a virtual machine. When operating in a virtual machine, it is preferable that the same installation modules be used. Using the same modules ensures that the testing environment matches the modules used in the real machine configuration. The only exception to this rule is the directory that appears on the virtual disk. The directory on the virtual disk cannot be the same as the real system directory because none of the labels and displacements for the user disks match.

To create the virtual directory for the virtual system residence volume, run CMS in the same virtual machine. First, set up a CMS file on one of the virtual disks. Second, have the virtual VM/SP user link to the CMS file with the desired filename and the filetype of DIRECT. The DIRECT program that is given control under CMS uses this file to create the virtual system's directory.

Note: The DIRECT file must contain sufficient directory entries to test VM/SP in a virtual machine environment.

Virtual IPL and Operation

Once a user has verified (by using a QUERY VIRTUAL command) that the virtual machine configuration matches or is a subset of the DMKRIO defined for the system to be tested, he can perform a virtual IPL of the virtual disk containing the CP nucleus. (In the example used at the beginning of this section, it is disk 330.) Because a terminal is handled like a simulated virtual console, (this example uses a 2741 terminal), each exclamation point (!) appearing in the sample terminal output indicates that the attention key has been pressed. The operation of the attention (ATTN) key on the terminal remains the same as it would have been if running any other system, but the operation of the virtual console is as though the device were an online console (3215) and not a 2741.

Note: Attention handling varies with the type of terminal used. Refer to the VM/SP Terminal User's Guide for a list of the terminals supported by VM/SP.

Proceed through the virtual machine IPL in the normal fashion, responding where required. Because the virtual VM/SP user cannot set the time-of-day clock, always reply "no" to the change time-of-day clock question. Under most circumstances, it is advisable to perform a cold start unless some specific function requiring a warm start is to be tested. When the test system has read/write access to a CMS minidisk, it can use the IPCS component of VM/SP to process any dumps taken of the virtual CP system. By using IPCS in the test system, an installation can standardize its VM/SP problem reporting and tracking process.

To place dumps of the virtual CP system in the test system's virtual reader:

1. Specify the test system's userid in the SYSDUMP operand of the SYSOPR system generation macro instruction.
2. Initialize the virtual CP system by assuming the SET DUMP AUTO CP ccommand (class B) by default.

Note: The test system's userid in the SYSDUMP operand should be OPERATOR, rather than the default of OPERATNS. OPERATOR helps the user to readily identify his dumps. It also makes the dump immediately available to the OPERATOR virtual machine user for IPCS processing.

Once the user IPLs the virtual machine and logs on the operator's virtual machine, he can operate virtual operating systems under this userid or enable virtual teleprocessing lines. Enabling these lines allows other users to dial into this system, logon to VM/SP in a virtual machine, and perform whatever actions they require.

ACCESSING DEVICES

Once a user IPLs the virtual machine, the devices that were not accessible to that machine at IPL time are considered offline. However, the user can attach more devices to this machine and have them placed online as required. For example: Tape drives can be attached by the real machine operator to the virtual machine configuration at the required address that matches the configuration of the virtual CP system. The same procedure can be used for teleprocessing lines, unit record equipment, or other devices.

Note: Most testing can be done by initializing and running tests from the operator's virtual machine without enabling any virtual teleprocessing lines.

Teleprocessing Lines and Spool Devices

Teleprocessing lines and spool unit record devices can be created by using the CP DEFINE command. Before the virtual CP operator can attach these lines or devices to a virtual machine user, they must first be placed online at the virtual machine level. Once online, they can be attached and used by virtual machines in the virtual CP system. (Teleprocessing lines can be attached directly to the virtual CP system for testing in that environment without using the CP DIAL command.)

Virtual Disks

It is possible to use virtual disks in the virtual CP system; however, their setup is complex and requires careful coordination with the real directory of the real system. For example: If a virtual disk is moved and the real directory of the real system is changed but the virtual directory is not changed, serious operating errors can occur; therefore, do not use virtual disks in the virtual CP system unless they are required for a specific test.

Note: When a virtual machine is linked to virtual disks before the user IPLs a system to run in the virtual machine, the virtual disks appear to the virtual system as disks with a zero relocation factor; that is, for CMS to access them at the virtual CP level, attach the disks at the CP level. Then the user can access them as though they were dedicated disks. Otherwise, accesses beyond this disk cause the real CP system to present I/O errors in the form of seek checks or command rejects to the virtual CP system, which, in turn, reflects the errors to the virtual operating system.

SPOOLING CONSIDERATIONS

If the virtual machine performs any spooling operations, the virtual CP system is also spooling unless it has dedicated unit record devices. This double spooling operation is not a problem; the virtual CP detects that it is running in a virtual machine and at the end of each spooled output file issues a CP CLOSE command to the real CP. This produces real spooled output for virtual spool files.

Also, note that double separators occur. For instance, the separator page on virtual printed output includes four pages (two pages for the virtual CP system and two more pages for the separator of the virtual machine on which the virtual CP system is running). The extra set of separator pages can be avoided by using the START command with the NOSEP option.

Because the virtual machine operation at this level is complex, there is no easy way of describing how to do all the functions. It requires careful study and analysis. At all times it requires an awareness of at what level of virtual machine is operating and what function the user is trying to perform.

Example -- Running VM/SP Under VM/SP

The following sample terminal session illustrates how to run a virtual CP system in a virtual machine environment. It is annotated to point out some of the more pertinent considerations.

```
| VM/370 online logo  
|  
| logcn v145r  
| ENTER PASSWORD:  
|  
| LOGMSG - 17:20:14 EDT THURSDAY mm/dd/yy |  
| * RUNNING SYS061--IPL 7  
| * QUERY LOG FOR RESTRICTIONS  
| LCGON AT 18:38:06 EDT THURSDAY mm/dd/yy |
```

This segment shows a normal logon procedure for a user identified as V145R. This userid is defined in the real CP directory with sufficient options and configurations to run VM/SP in a virtual machine environment.

```
| query virtual  
| STORAGE = 00512K  
| RDR 00C CLS A  
| PUN 00D CLS A COPY 01  
| PRT 00E CLS A COPY 01  
| CONS 01F ON DEV 051  
| DASD 190 2314 CMS370 R/O 056 CYL  
| DASD 19A 2314 CMS190 R/O 055 CYL  
| DASD 19E 2314 CMS190 R/O 026 CYL  
| DASD 290 2314 PDISK3 R/O 045 CYL  
| DASD 330 3330 PDISK4 R/W 020 CYL
```

After logon, issuing the QUERY VIRTUAL command permits a user to verify the virtual machine configuration. The response indicates:

- The storage size is 512K bytes.
- Some unit record devices have been defined.
- The console is 01F and is real device 051.
- Devices 190, 19A, and 19E are used to operate CMS in this virtual machine.
- Device 290 is not used and could have been deleted.
- Device 330 is the 3330, 20-cylinder, read/write minidisk that becomes the virtual system residence volume for this virtual system when it is running VM/SP.
- The volume serial numbers (volids) for the DASD units are those of the real disks on the real computing system.

```
| link usecms 191 191 rr  
| ENTER READ PASSWORD:  
|  
| DASD 191 LINKED R/O; R/W BY USECMS
```

The LINK command allows the user to access a userid that has a CMS disk containing certain directory files. This example uses one of these files to create a virtual system directory.

```
| def 1f as 009  
| CONS 009 DEFINED  
| i cms  
| CMS...FLOOR...mm/dd/yy  
|  
| Y (19E) R/O.  
| A (191) R/O.  
| 013 USERS, 000 DIALED  
| DMSACC113S 'B (196) ' NOT ATTACHED.  
| DMSACC113S 'C (194) ' NOT ATTACHED.  
| R;
```

This segment shows the redefinition of console 01F as 009 before issuing the CP IPL command. The error messages indicate that a PROFILE EXEC is running from the user's 191 disk; the PROFILE EXEC is attempting to access disks that are not defined in the virtual machine configuration. However, these disks are not required to do a directory load.

```
| listf * direct a  
| FILENAME FILETYPE MODE |  
| V145A     DIRECT    A1  |  
| USERTEST  DIRECT    A2  |  
| USER     DIRECT    A1  |  
| USER1    DIRECT    A1  |  
| R;
```

This LISTFILE command, issued in the CMS environment, shows that there are four files with a filetype of DIRECT. This example uses the one named V145A.

Example -- Running VM/SP Under VM/SP

The following sample terminal session illustrates how to run a virtual CP system in a virtual machine environment. It is annotated to point out some of the more pertinent considerations.

```
| VM/370 online logo  
|  
| logcn v145r  
| ENTER PASSWORD:  
|  
| LOGMSG - 17:20:14 EDT THURSDAY mm/dd/yy |  
| * RUNNING SYS061--IPL 7  
| * QUERY LOG FOR RESTRICTIONS  
| LCGON AT 18:38:06 EDT THURSDAY mm/dd/yy |
```

This segment shows a normal logon procedure for a user identified as V145R. This userid is defined in the real CP directory with sufficient options and configurations to run VM/SP in a virtual machine environment.

```
| query virtual  
| STORAGE = 00512K  
| RDR 00C CLS A  
| PUN 00D CLS A COPY 01  
| PRT 00E CLS A COPY 01  
| CONS 01F ON DEV 051  
| DASD 190 2314 CMS370 R/O 056 CYL  
| DASD 19A 2314 CMS190 R/O 055 CYL  
| DASD 19E 2314 CMS190 R/O 026 CYL  
| DASD 290 2314 PIDSK3 R/O 045 CYL  
| DASD 330 3330 PIDSK4 R/W 020 CYL
```

After logon, issuing the QUERY VIRTUAL command permits a user to verify the virtual machine configuration. The response indicates:

- The storage size is 512K bytes.
- Some unit record devices have been defined.
- The console is 01F and is real device 051.
- Devices 190, 19A, and 19E are used to operate CMS in this virtual machine.
- Device 290 is not used and could have been deleted.
- Device 330 is the 3330, 20-cylinder, read/write minidisk that becomes the virtual system residence volume for this virtual system when it is running VM/SP.
- The volume serial numbers (volids) for the DASD units are those of the real disks on the real computing system.

```
| link usecms 191 191 rr  
| ENTER READ PASSWORD:  
|  
| DASD 191 LINKED R/O; R/W BY USECMS
```


The LINK command allows the user to access a userid that has a CMS disk containing certain directory files. This example uses one of these files to create a virtual system directory.

```
def 1f as 009
CONS 009 DEFINED
i cms
CMS...FLOOR...mm/dd/yy

Y (19E) R/O.
A (191) R/O.
013 USERS, 000 DIALED
DMSACC113S 'B (196) ' NOT ATTACHED.
DMSACC113S 'C (194) ' NOT ATTACHED.
R;
```

This segment shows the redefinition of console 01F as 009 before issuing the CP IPL command. The error messages indicate that a PROFILE EXEC is running from the user's 191 disk; the PROFILE EXEC is attempting to access disks that are not defined in the virtual machine configuration. However, these disks are not required to do a directory load.

```
listf * direct a
FILENAME FILETYPE MODE
V145A DIRECT A1
USERTEST DIRECT A2
USER DIRECT A1
USER1 DIRECT A1
R;
```

This LISTFILE command, issued in the CMS environment, shows that there are four files with a filetype of DIRECT. This example uses the one named V145A.

```

| type v145a direct
|
| DIRECTORY 330 3330 CPDSK3
|   USER OPERATOR OPERATOR 256K 1M ABCDEFG
|   ACCOUNT 12345678 COMP.RM
|     CONSOLE 9 3215
|     SPOOL C 2540 READER A
|     SPOOL D 2540 PUNCH A
|     SPOOL E 1403 A
|     LINK USECMS 191 191 WR
|     MDISK 196 3330 0 10 SYS196 RR RDGDEV
|     MDISK 190 2314 0 56 CMS190 RR RDGDEV
|     MDISK 19E 2314 0 26 FLRCMS RR RDGDEV
|   USER USECMS TOM 256K 1M G
|   ACCOUNT 12345679 ROOM331
|     CONSOLE 9 3215
|     SPOOL C 2540 READER A
|     SPOOL D 2540 PUNCH A
|     SPOOL E 1403 A
|     LINK OPERATOR 196 196 RR
|     LINK OPERATOR 190 190 RR
|     LINK OPERATOR 19E 19E RR
|     MDISK 191 3330 0 9 USECMS WR RDGDEV WDGDEV MDGDEV
|     MDISK 192 2314 T-DISK 5
|     DED 19A CMS19A
|
| R;

```

This segment requests a typed copy of file V145A DIRECT. The DIRECTORY statement specifies that the directory is to be written on a 3330 device at address 330 and that its virtual label is CPDSK3. This address corresponds to the 3330 virtual disk that was shown and discussed under the QUERY VIRTUAL command at the beginning of this example. Because this is a virtual disk, CPDSK3 is a virtual label, not a real label. Thus, the virtual 3330 disk (CPDSK3) is on the real disk labeled PDISK4. This virtual disk was previously formatted, labeled, and allocated by the CP format/allocate program (not shown in this example).

Note: The user identified as OPERATOR has all privilege classes to easily control the virtual VM/SP system. The console and unit record devices are defined to allow him to operate CMS. The virtual disks defined for this userid have a displacement of zero and a size that does not exceed the bounds of the virtual disks defined for the virtual VM/SP system. The volids specified on the directory statements are the volids on the virtual disks for the virtual CP system. They are not the volids of the real disks on which those virtual disks are defined for the virtual CP system.

```

| direct v145a
| EOJ DIRECTORY UPDATED
| R(00006);

```

This segment shows the operation of the directory program in a virtual machine. The file used to create the virtual directory is V145A DIRECT, which was previously typed out. Note that the return code is 6. The directory has been updated on the disk, but because this disk is a virtual disk and not the real system residence disk, the real CP system directory has not been modified. The return code of 6 is the normal code indicating this fact. However, a return code of 4, 5, or 6 is acceptable.

```
| det 191  
| DASD 191 DETACHED |  
| R; |
```

Since the 191 disk of user USECMS is no longer needed, it is now detached.

```
| link cpsys 196 196 rr  
| ENTER READ PASSWORD:  
  
| R;  
  
| link cpsys 194 194 rr  
| ENTER READ PASSWORD:  
  
| R;  
  
| acc 196 a  
| '196' REPLACES ' A (191) '  
| A (196) R/O.  
| R;  
  
| acc 194 b/a  
| B (194) R/O.  
| R;
```

Before doing a virtual VMFLOAD function, it is necessary to access the disks required to perform this function. The LINK commands define the disks that contain the CP system to be tested in a virtual machine environment. The CMS ACCESS commands access those disks and place them in a read-only status.

```
| spccl pun * |  
| R; |
```

The CP SPOOL command transfers the output of the spool punch back to this userid. This is required so that the user may later IPL the virtual card reader to load the CP nucleus onto the virtual system residence disk.

```
| spccl prt * |  
| R; |
```

The CP SPOOL command transfers the output of the printer back to this userid. This transfer is required so that the user may later read in the nucleus load map used by IPCS for creating problem reports.

```
| vmfload cplcad ptmx  
| *****SYSTEM LOAD DECK COMPLETE  
| PCH FILE 0189 TO V145R  
| R;
```

The VMFLOAD function is executed specifying the load list of CLOAD and a control file of PTMX. PTMX is a special control file used to apply experimental updates and PTFs. The series of asterisks are the CMS blip character, which CMS types or displays whenever the virtual machine uses two seconds of processor time. At the completion of the load function, the spool file is transferred to V145R and is available as a reader file.

```

!!
CP
ipl 00c
NUCLEUS LOADED ON CPDSK3
DMKDSP450W CP ENTERED; DISABLED WAIT PSW
CP

```

The user is now finished using CMS for the directory and IPL deck setup. The IPL of card reader 00C loads the nucleus, and the loader is distributed with the following default I/O addresses:

```

CCNSOLE = 009
PRINTER = 00E

```

```

clse prt
PRT FILE 0190 TO V145R COPY 01 NOHOLD

```

The CLOSE command causes VM/SP to place the nucleus load map in the virtual reader. The following message indicates that this nucleus has been loaded on the virtual disk.

```

NUCLEUS LOADED ON CPDSK3

```

The virtual minidisk label must either be CPDSK3 or be defined in the DMKSYS module. The virtual machine enters the disabled wait state after producing a message from the real CP system.

```

def 009 as 01f
CONS 01F DEFINED

```

The console must be redefined as 01F. This is the console address that was specified in DMKRIO which was loaded as part of the CP nucleus.

```

i cms
CMS . . . FLOOR . . . mm/dd/yy
Y (19E) R/O
R;
read testnuc map
RECORD LENGTH IS '132' BYTES
R;

```

Initializing CMS and reading TESTNUC MAP places the nucleus load map on the test userid's CMS A-disk. By saving the load map on this disk, the test system at the virtual CP level can use IPCS to access it and create problem reports.

```

| query virtual
| STORAGE = 00512K
| RDR 00C CLS A
| PUN 00D CLS A          TO V145R
| PRT 00E CLS A          COPY 01
| CCNS 01F ON DEV 051
| DASD 190 2314 CMS370 R/O 056 CYL
| DASD 191 2314 PIDSK3 R/W 010 CYL
| DASD 194 3330 PIDSK5 R/O 060 CYL
| DASD 196 3330 PIDSK7 R/O 010 CYL
| DASD 19A 2314 CMS190 R/O 055 CYL
| DASD 19E 2314 CMS190 R/O 026 CYL
| DASD 290 2314 PIDSK3 R/O 045 CYL
| DASD 330 3330 PIDSK4 R/W 020 CYL

```

The QUERY VIRTUAL command displays the current virtual machine configuration. This is the configuration that was used to run the CMS machine, except that the console address has been changed to 01F. Before initializing the virtual 330 disk and bringing in VM/SP, it is necessary to redefine the disk addresses so that they can be recognized by the VM/SP system.

```

| define 190 as 130
| DASD 130 DEFINED
| define 194 as 331
| DASD 331 DEFINED
| define 196 as 332
| DASD 332 DEFINED
| define 19e as 131
| DASD 131 DEFINED
| link virttest 191 333 r
| ENTER READ PASSWORD:
|
| DASD 333 LINKED R/O

```

These DEFINE commands and the LINK command change the configuration of the virtual machine so that it can be recognized by the virtual VM/SP nucleus. Note that the 2314s are defined in the 2314 range of 130 to 137 and that the 3330s are defined in the 3314 range of 130 to 137 and that the 3330s are defined in the 3330 range of 330 to 337. The LINK command is used to access another user's disk as a 3330 at address 333.

```

| query virtual
| STORAGE = 00512K
| RDR 00C CLS A
| PUN 00D CLS A          TO V145R
| PRT 00E CLS A          COPY 01
| CCNS 01F ON DEV 051
| DASD 130 2314 CMS370 R/O 056 CYL
| DASD 131 2314 CMS190 R/O 026 CYL
| DASD 19A 2314 CMS190 R/O 055 CYL
| DASD 290 2314 PIDSK3 R/O 045 CYL
| DASD 330 3330 PIDSK4 R/W 020 CYL
| DASD 331 3330 PIDSK5 R/O 060 CYL
| DASD 332 3330 PIDSK7 R/O 010 CYL
| DASD 333 3330 PIDSK7 R/O 010 CYL

```

A CP QUERY VIRTUAL command is issued again to show that the virtual machine configuration has been redefined to match the one that can be recognized by the virtual VM/SP system. Notice that the 330 disk has read/write status (this is required for VM/SP to do virtual paging and spooling). All the other disks have read-only status. Disks 19A and 290 are not recognized by the virtual VM/SP system because they are not defined in the DMKRIO module; however, their inclusion the configuration does not in the configuration does not matter.

```

-----
| ipl 330 |
-----

```

The virtual VM/SP system is loaded by an IPL of the virtual system residence volume (330).

```

-----
| VM/SP VERSION x LEVEL 1 SL nnn mm/dd/yy hh:mm:ss
|
| NOW 19:02:54 EDT THURSDAY mm/dd/yy
| CHANGE TOD CLOCK (YES|NO) :no
| 19:03:15 DMKLNK018E USECMS 191 NOT LINKED; VOLID USECMS NOT MOUNTED
| RRRR....RING....GGGG
| 19:03:16 DMKLNK108E OPERATOR 19E NOT LINKED; VOLID FLRCMS NOT MOUNTED
| RRRR....RING....GGGG
| 19:03:16 LOGON AT 19:03:16 EDT THURSDAY mm/dd/yy
| 19:03:16 LINE 01F LOGON AS OPERATOR USERS = 001
| 19:03:16
-----

```

This output is from the VM/SP system running in a virtual machine. It is printing the responses on what appears to it as a virtual 3215 console. Note that the CHANGE TOD CLOCK (YES/NO) prompting message does not require a response. If the response had been yes, it would have requested a date and time to be set; however, the real time-of-day clock cannot be changed from a virtual machine environment. The LINK error messages are a result of the automatic operator logon and of the directory not being able to find some disks defined in the operator's virtual machine. The "RING" message is the real CP simulation of the virtual console alarm function. Finally, the operator receives confirmation of a logon.

```

-----
| DMKCPI951I CP VOLID CPDRM1 NOT MOUNTED
|
| RRRR....RING....GGGG
| 19:03:16
|
| DMKCPI951I CP VOLID PIDSK2 NOT MOUNTED
|
| RRRR....RING....GGGG
| 19:03:16 START ((COLD|WARM) (DRAIN))|(SHUTDOWN) :cold
| 19:04:00 FILES: NO RDR, NO PRT, NO PUN
-----

```

VM/SP issues the messages indicating that CPDRM1 and PIDSK2 are not mounted because the virtual DMKSYS module has an owned list that has three volumes specified: CPDRM1, PIDSK2, and PIDSK3. The only one available in the configuration during IPL was the system residence volume, CPDSK3. These error messages are not severe: only a minimum amount of space is required by CP to accomplish paging and spooling. The response to the start message in this case is "cold," which should be the normal response unless a specific test of warm start is required.

```

!
19:04:23 query dasd all
19:04:30 DASD 130 CP SYSTEM CMS190    001
19:04:30 DASD 131 FREE
19:04:30 DASD 132 OFFLINE
19:04:30 DASD 133 OFFLINE
19:04:30 DASD 134 OFFLINE
19:04:30 DASD 135 OFFLINE
19:04:30 DASD 136 OFFLINE
19:04:30 DASD 137 OFFLINE
19:04:30 DASD 250 OFFLINE
19:04:30 DASD 251 OFFLINE
19:04:30 DASD 252 OFFLINE
19:04:30 DASD 253 OFFLINE
19:04:30 DASD 254 OFFLINE
19:04:30 DASD 255 OFFLINE
19:04:30 DASD 256 OFFLINE
19:04:30 DASD 257 OFFLINE
19:04:30 DASD 2D0 OFFLINE
19:04:30 DASD 2D1 OFFLINE
19:04:30 DASD 2D2 OFFLINE
19:04:30 DASD 330 CP OWNED  PIDSK4    001
19:04:30 DASD 331 CP SYSTEM CPRL10    001
19:04:30 DASD 332 CP SYSTEM SYS196    001
19:04:30 DASD 333 FREE
19:04:30 DASD 334 OFFLINE
19:04:30 DASD 335 OFFLINE
19:04:30 DASD 336 OFFLINE
19:04:30 DASD 337 OFFLINE
19:04:30 DASD 350 OFFLINE
19:04:30 DASD 351 OFFLINE
19:04:30 DASD 352 OFFLINE
19:04:30 DASD 353 OFFLINE

```

In this example: The exclamation mark (!) indicates that an attention has been signaled. VM/SP reflects this signal as an attention to the virtual machine. The virtual CP system responds to the attention by typing the time and issuing a read. The response to the read is the entry of the QUERY DASD command. The response to this command from the virtual CP system is the DASD status shown. Notice that most of the devices are in an offline condition, since at the time of the IPL these device addresses were not available in the virtual machine configuration. The devices that were available are now marked free, owned, or system. (The system volumes are ones that have minidisks in use by the operator.) Notice that device 332 has a label of SYS196 in the virtual CP system. A previous QUERY VIRTUAL showed that DASD 332 is actually physically mounted on PIDSK7. However, this label is the real system label and is not the one recognized by the virtual CP system. For users to access the 332 disk, the virtual directory must refer to the virtual label of SYS196. (The operator's virtual disk 196 refers to a zero cylinder displacement on volume SYS196.)

```

!
19:06:34 q virtual
19:06:40 STORAGE = 00256K
19:06:40 CONS 009 ON DEV 01P
19:06:40 RDR 00C CLS A
19:06:40 PUN 00D CLS A          COPY 01
19:06:40 PRT 00E CLS A          COPY 01
19:06:40 DASD 190 2314 CMS190 R/O 056 CYL
19:06:40 DASD 196 3330 SYS196 R/O 010 CYL

```

Again, the user signals attention to the virtual CP system. The operator types in QUERY VIRTUAL, and the display is the virtual machine configuration for the virtual machine operator. Note that the operator has a configuration that is suitable for running CMS by loading (via IPL) virtual device 190.

```
| 19:07:38 att 131 operator 191 |
| 19:07:55 DASD 131 ATTACH TO OPERATOR 191 |
```

The operator attaches what appears to him as real disk 131 to himself as virtual address 191. The response indicates a successful attach.

```
| !! |
| CP |
| q v 131 |
| DASD 131 2314 CMS190 R/O 026 CYL |
```

The attention signalled here, shown by two exclamation marks followed by the word CP, indicates that the user is at the real CP level. At that level, issue a QUERY VIRTUAL 131. The response indicates that the virtual 131 disk is a 2314 with read-only status, with 26 usable cylinders.

```
| ! |
| 19:08:40 q 131 |
| 19:08:50 DASD 131 ATTACH TO OPERATOR 191 |
| ! |
| 19:08:58 q v 191 |
| 19:09:04 DASD 191 ON DEV 131 |
```

Signalling attention takes the user back to the virtual machine level, where an attention interrupt is reflected. The virtual CP system then responds with the time and issues a read. At the virtual CP system level, the user issues a QUERY 131, which for the operator is a query of what appears to him as real disk 131. Note that the status is that of the disk attached to the operator as virtual address 191. This is the same disk that was previously noted; however, the virtual CP system thinks that the disk has read/write status. The single attention again causes a read, and the user issues a QUERY VIRTUAL 191. The response indicates a dedicated disk on device 131 and assumed read/write status.

```
| ! |
| 19:09:16 ipl 190 |
| 19:09:23 CMS..FLOOR..mm/dd/yy |
| DMSACC112S 'A (191) ' DEVICE ERROR |
| R; T=0.02/0.04 19:11:29 |
```


Signalling attention again causes a CP read, and the operator performs a virtual IPL of the virtual 190 disk to load the CMS system. The response is from the CMS system running in a virtual machine under a virtual VM/SP system running under a real VM/SP system. A carriage return to the ensuing read gives an error message from CMS. The error message appears because CMS has an indication from the virtual CP system that it has write access to the disk (since it appears as a dedicated disk). However, the real CP system has the disk in read-only status and rejects the write attempt to the virtual CP system, which in turn reflects it to CMS, causing the device error message.

```

!!
| CP
| det 333
| DASD 333 DETACHED
| link virttest 191 333 w
| ENTER WRITE PASSWORD:
|
| DASD 333 LINKED R/W

```

The user then enters the real CP mode by signalling attention. The user detaches device 333 and links to it as 333 in write mode. The fact that the operator detached and relinked is transparent to the virtual CP system at this level. The user has accomplished a status change from read to write. The physical extent definition has not changed.

```

!
| 19:15:38 det 191#att 333 operator 191
| 19:15:52 DASD 131 DETACHED OPERATOR 191
| 19:15:52 DASD 191 DETACHED
| 19:15:43 DASD 333 ATTACH TO OPERATOR 191
| 19:15:53 b
| CMS
|
| acc 191 a
| R; T=0.39/0.76 19:16:23

```

Signalling attention causes a read from the virtual CP system, where the operator detaches the virtual 191 disk and attaches the real 333 disk to his userid as 191. Note that the 333 appears to the virtual CP system as a real disk, when it actually is a virtual disk. The BEGIN command changes the virtual machine environment to CMS. The ACCESS 191 command is then successfully completed, giving write access to the virtual 191 disk, which is the virtual CP system's 333 disk previously linked in write mode.

```

| print profile exec
| 19:16:45 PRT 00E OUTPUT OF OPERATOR FILE = 0002 LINES= 00013
| R; T=0.23/0.51 19:16:46
|
| !
| 19:17:05 drain 00e
| 19:17:04 PRT 00E SPOOL CLS XA DRAINED

```

From CMS, the PROFILE EXEC is printed. The virtual CP system responds with a printer output message for file 2, which is the output from the previous print function. The ready message is the response from the CMS system. This example shows a virtual machine running with a virtual console that is receiving both virtual machine and CP messages. Signaling attention places the virtual machine in virtual CP mode, where the user can specify a drain of device 00E. The system responds with a message indicating that the device is drained. This indicates that the virtual CP system has completed printing on what it thinks is a real printer. This printer is actually spooled by the real CP system.

```

| !
| 19:19:44 set dump auto CP
| !
| 19:19:51 g dump
| 19:19:55 DASD 330 DUMP UNIT CP

```

Signalling attention takes the user to the virtual CP level, where he issues a SET DUMP command. Ordinarily, when testing an unstable system, this would have been one of the first commands entered after issuing the IPL for the virtual CP system. The query of the dump unit verifies that the dump is of the CP nucleus to the spooling disk at address 330.

```

| !!
| CP
| system restart
| RRRR....RING....GGGG
| 19:20:06 DMKDMP908I SYSTEM FAILURE; CODE PSA002
| RRRR....RING....GGGG
| RRRR....RING....GGGG
| DMKCKP960I SYSTEM WARM START DATA SAVED
| DMKDSP450W CP ENTERED; DISABLED WAIT PSW
| CP
| DMKCKP961W SYSTEM SHUTDOWN COMPLETE
| CP
| RPRR....RING....GGGG
| CP

```

Signalling attention takes the user to the real CP level, where he enters the command SYSTEM RESTART. This command is the equivalent of a system restart function on a real processor. The system restart function for a CP system automatically dumps the system and then issues IPL again. After the system is dumped, a message appears with abnormal termination code PSA002 (a system dump due to pressing the system restart key).

The virtual bell rings to indicate that the system has been reloaded, and the system prints messages about: saving warm start data, CP entering a disabled wait state, and system shutdown being complete. The message indicating that CP has entered a disabled wait state is prematurely issued between these two messages. It occurs because of a synchronization of the real CP system with the virtual CP system console output.

After these messages are issued, the user is in real CP mode. He can either log off or obtain the system abend dump.

To obtain this dump, re-IPL 330 and repeat the test procedure up to the point where the 'print profile exec' is shown in the same session. At this point, the user now has CMS initialized in the virtual CP system and has read/write access to his real CMS minidisk at virtual address 191. By issuing a QUERY RDR ALL command, VM/SP should reveal that a class D dump file is in the operator's virtual reader (because the operator's userid is specified in the SYSDUMP operand of the SYSOPR system generation macro instruction).

```
-----  
| !  
| 19:22:10 spool rdr cl d |  
| 19:22:10 b             |  
| CMS                    |  
| vmfddump               |  
| .                      |  
| .                      |  
| .                      |  
| R;                     |  
-----
```

The SPOOL RDR CL D command allows the virtual reader to access the class D dump file. By entering the B (BEGIN) command, the virtual machine user returns to CMS. By entering the VMFDUMP command (class C or E), IPCS reads the CP abend dump and creates a CMS dump file, problem report, and symptom summary entry on the 191 A-disk. (For a sample VMFDUMP session, refer to VM/370 IPCS User's Guide.)

When VMFDUMP processing completes under CMS in the virtual CP system, terminate the test system by entering real CP mode and initializing CMS. Under CMS, the user can issue the IPCS DUMPSCAN command to look at the dump taken of his virtual test system. This dump resides as a dump file on the user's real 191 A-disk.

Summary

To update and test a VM/SP system in a virtual machine, an installation must first have a VM/SP directory entry for a VM/SP virtual machine. This virtual directory entry need only specify the minimum number of users sufficient to perform the test. Before initializing this system, an installation should verify that the virtual machine configuration has: the correct console address, sufficient unit record devices available at the correct addresses, and enough disks (either linked or attached) to make a reasonable test. Also, the VM/SP virtual machine can run CMS when it has access to the CMS system residence volume.

With few exceptions, IPL for a VM/SP virtual machine is similar to IPL for a real VM/SP system. Operationally, VM/SP provides CP commands to display and store into real storage. The VM/SP system in a virtual machine can also display and store into its own third level virtual storage. If the virtual machine performs any spooling operations, the virtual VM/SP system is also spooling unless it has dedicated unit record devices. This double spooling is no problem; however, there are some special operational considerations.

Section 3. DOS/VS in a Virtual Machine

This section uses the term "DOS/VS" as a generic expression. It represents any or all of the DOS, DOS/VS, and VSE environment system control programs unless specified otherwise.

When loading DOS/VS into a virtual machine running under VM/SP, the terminal becomes the DOS/VS operator console, and the user is responsible for entering all the commands and responses normally required of the operator.

The four basic techniques to use when running DOS/VS in a virtual machine are:

- Run DOS/VS in batch mode. The terminal is the operator console, and other users may submit jobs either through the real system virtual card reader or from the virtual card punches of other userids.
- Use the JPL command to alternate between using DOS/VS and CMS in a single virtual machine. Use CMS to prepare a job stream for DOS/VS, use DOS/VS to execute the job stream, and use CMS to check the output.
- Logon to a userid and load DOS/VS. Once it is running, disconnect from that userid and logon to another userid that has been established in the CP directory as the secondary user for the first user. Then both virtual machines can be operated from one terminal.
- Logon to a userid and load DOS/VS. Once it is running, disconnect from that userid and logon to another userid while the DOS/VS userid continues working. To check on the status of DOS/VS, disconnect from the current virtual machine and reconnect the DOS/VS virtual machine.

Before discussing these four techniques in greater detail, the user must understand how to:

- Generate DOS/VS to run in a virtual machine
- Create VM/SP directory entries for DOS/VS virtual machines
- Access the DOS/VS system residence volume
- Ensure that the proper I/O devices are attached to the DOS/VS virtual machine
- IPL and operate DOS/VS under VM/SP

Note: Multiple DOS/VS statements cannot be entered on a single line using the logical line end character (#). All logical line end characters translate to a X'15' before being passed to a virtual machine; DOS/VS does not recognize this condition.

System Generation Recommendations

When generating DOS/VS to run in a virtual machine, an installation should have these primary objectives:

- To reduce the number of SIO instructions issued by DOS/VS, including those issued for DOS/VS paging I/O operations
- To avoid double CCW translation

To meet these objectives, an installation needs to consider how it generates both VM/SP and DOS/VS.

Note: The following recommendations have been made by users who run DOS/VS in a virtual machine under VM/SP. As such, these recommendations have not been submitted to any formal IBM tests. Prior to any implementation, an installation should evaluate their usefulness in its own configuration.

VM/SP RECOMMENDATIONS

When generating VM/SP for a DOS/VS virtual machine, note the following recommendations:

VM/SP Saved Systems

IPL time can be reduced by saving any operating system after the generated operating system has been loaded on VM/SP. For more information about generating saved systems, refer to the VM/SP System Programmer's Guide.

Handshaking for DOS/VS

DOS/VS Release 34 with the Advanced Functions-DOS/VS Program Product (Program No. 5746-XE2) uses VM/VS handshaking. For further details, refer to the appropriate DOS/VS program product publications. DOS/VSE with the VSE/Advanced Functions Program Product (5746-XE8) uses VM/VS handshaking (also known as the DOS/VSE-VM/370 linkage facility). For further details, refer to VSE/Advanced Functions General Information, GC33-6106.

Initializing Disks and Minidisks

To initialize a DOS/VS minidisk for use under VM/SP, the VM/SP IBCDASDI service program must be used. If the whole disk is used by DOS/VS, any DOS or OS disk initialization program may be used.

DOS/VS RECOMMENDATIONS

When generating DOS/VS to run in a virtual machine, note the following recommendations:

Generating a DOS/VS Supervisor

- Specifying boundary size (DOS/VS Release 34 and earlier only)
When generating a DOS/VS supervisor, use 4K (the size of VM/SP's pages) whenever DOS/VS recommends using a 2K boundary or a multiple of 2K. Thus, do not use the default for the SEND macro instruction. It causes DOS/VS to round the supervisor size to the next 2K boundary. Instead, manually calculate the size of the supervisor and specify a 4K boundary in the SEND macro instruction. This specification forces DOS/VS to be loaded at the next 4K page boundary.
- Specifying console addresses
Specify an address of 01F in the VM/SP directory CONSOLE statement. This specification allows the DOS/VS supervisor to run on the real processor using the real console address of 01F.

If CMS is also to be used, it uses console address 01F. In this case, specifying 01F in the CONSOLE statement allows use of this console with both DOS/VS and CMS virtual machines.

To use more than one DOS/VS console under VM/SP or to use a 3270 display device in display operator control mode, refer to the topic "Using a Display Terminal as a Console in Display Mode" in the "General Considerations" section of this publication.

- Reducing privileged instructions
Improve performance by reducing the number of privileged instructions that must be handled by VM/SP and virtual machine assist. Generate a tailored DOS/VS supervisor for each virtual machine and leave out any unnecessary options. Because VM/SP issues its own stand-alone seek (except for 2314 disks), do not specify seek separation in the FOPT macro instruction.

When using DOS/VS Release 34 (or earlier) and the processor has block multiplexer channels, specify the block multiplexer option in both the PIOCS macro instruction and the VM/SP directory OPTION statement. However, this specification is unnecessary when using the VSE environment because block multiplexer support is standard.

Generate several tailored DOS/VS supervisors, if desired, and store them on the same DOS/VS SYSRES with different supervisor names (such as \$\$A\$\$SUPn, where n is 1, 2, 3, 4, etc.).

Specifying DOS/VS Partition Sizes (DOS/VS Release 34 and earlier only)

Performance is usually better when using several virtual machines rather than using many active partitions in one virtual machine. That is, if there is a communications system, a batch system, and a test system, create a separate virtual machine for each one. However, to only run VM/SP part of the day and to minimize operational differences, one multipartition production DOS/VS machine may be preferable.

It is usually best to make the DOS/VS partition sizes, and therefore the whole DOS/VS virtual machine, large enough so that all jobs run V=R. Let VM/SP do the paging.

Rotational position sensing (RPS) cannot be used with V=R partitions. Also, avoid double CCW translation and double paging.

Set RSIZE equal to the supervisor size plus the sum of all V=R partitions, plus the SVA, plus 32K.

Note: An installation must specify ",REAL" on the DOS/VS // EXEC job control card.

Generating the Operating System

When VM/SP is the primary operating system and DOS/VS is running one or two partitions in a virtual machine, generate DOS/VS with as few options as possible, particularly when several virtual machines share the same system residence volume.

When VM/SP is not the primary operating system and DOS/VS is usually run without VM/SP, generate DOS/VS to:

- Be transparent to the users of the other systems
- Have the required number of partitions

Generating POWER/VS or VSE/POWER

When DOS/VS is run under VM/SP, POWER (POWER/VS for DOS/VS or VSE/POWER for the VSE environment) should be used with the appropriate unit record devices that are dedicated to DOS/VS.

If an installation has sufficient DASD space, let both POWER and VM/SP spool. Generate POWER with only the options that suit the installation's needs, but make the I/O buffer sizes as large as possible, up to 2008 bytes. If one job step in a DOS/VS job stream abends, it is easy to use POWER to cancel the remainder of the job stream. To use only VM/SP spooling, an installation must manually cancel each job step.

Sharing the DOS/VS System Residence Volume

There are two methods of sharing a DOS/VS system residence volume:

- Have a read/write DOS/VS system residence (SYSRES) minidisk for each DOS/VS virtual machine. Then share a read-only copy of the private core image, relocatable, and source statement libraries.
- Share a read-only copy of the DOS/VS SYSRES minidisk, and have separate read/write private libraries for each virtual machine. Because only one standard label cylinder is available, all virtual machines must coordinate their use of that cylinder.

Which method is used depends upon each installation's specific situation.

Restoring the DOS/VS PID Tape

When generating DOS/VS under VM/SP, an installation must restore the DOS/VS PID tape onto a full disk pack; a minidisk cannot be used. Create an operational system residence pack on a full pack, and then use CORGZ to copy the DOS/VS SYSRES to a minidisk.

Reducing SLD Directory Reading

In the FOPT macro instruction, specify enough second level directory (SLD) entries to reduce repetitious reading of the directory.

Use the system directory list (SDL) in the shared virtual area (SVA) for all job control, disk and tape open and close transients, and for the attention routine.

Selecting File Names for Minidisks

If minidisks are used, although permitted on DOS/VS running native, do not use the same DOS/VS file name for more than one disk. Otherwise, VM/SP may select the wrong minidisk.

Executing Programs Under DOS/VS and CMS/DOS

If a program in the DOS/VS core image library may be executed in either a DOS/VS virtual machine or under CMS/DOS, link edit the program with ACTION REL linkage editor control statement so that it uses the DOS relocating loader.

DOS/VS ACCOUNTING

Note: This topic only applies when running DOS/VS Release 34 (or earlier). It does not apply when running either DOS/VS Release 34 with the Advanced Functions-DOS/VS Program Product (5746-XE2) or running in the VSE environment.

Except with processors that have ECPS: VM/370 and virtual interval timer assist, DOS/VS accounting gives inaccurate and inconsistent elapsed processor times when operating under VM/SP with virtual machine assist. This inconsistency occurs because the interval timer (located at virtual storage location X'50') used by the DOS/VS accounting routine is only updated when VM/SP gets control. Therefore, when DCS/VS accesses the interval timer data, a variable amount of time may have elapsed since VM/SP last updated the interval timer, and thus DOS/VS records an inaccurate processor time.

To attempt to minimize this inaccuracy at the cost of some additional VM/SP overhead, an installation may wish to add the following dummy DIAGNOSE instruction:

83000000

at the following locations in the DOS/VS supervisor source statements:

- In the SVC 24 routine, before the L R3,SYSTIMER statement
- In the SVC 52 routine, before the L R3,SYSTIMER statement
- In the STCLOCK routine, before the STCK CLOCK statement
- In the timer interruption handler routine, before the LM R2,R3,SYSTIMER statement
- In the job accounting initialization routine (JATIMER), before the statement that references SYSTIMER

Note: To prevent a possible specification exception to DOS/VS, ensure that general register zero contains zeros before issuing the DIAGNOSE instruction.

If VM/SP is running on a processor model that has ECPS: VM/370 (as defined in VM/SP Planning and System Generation Guide), enable virtual interval timer assist. This action lets the hardware, rather than VM/SP, update the virtual interval timer. Hardware update frequency is 300 times per second and results in accurate and repeatable time measurements.

Generating DOS/VS Under VM/SP

This topic presents the major steps in a DOS/VS system generation procedure that is performed under VM/SP. This virtual machine is assumed to have a VM/SP directory entry as shown in Figure 14. The link to CMSSYS provides the user with CMS facilities. Alternate procedures to CMS are discussed wherever applicable.

The virtual machine console is assumed to be a 3270 display terminal.

```
-----  
| USER DOSVSYSPASSWORD 512K  
| ACCOUNT ACCTNO BIN4  
| OPTION ECMODE  
| CONSOLE 009 3210  
| SPOOL 00C 2540 R  
| SPOOL 00D 2540 P  
| SPOOL 00E 1403  
| LINK CMSSYS 190 190 RR  
| MDISK 191 3330 50 5 UDISK3 MR  
| MDISK 250 3330 0 404 DOSSYS MR  
|-----
```

Figure 14. Virtual Machine for DOS/VS System Generation

In addition to the devices specified in the directory entry for user DOSVSYSP, this user temporarily needs a dedicated magnetic tape unit at address 180 on which to mount the distribution tape. The attachment of the magnetic tape drive is discussed later in this topic.

BUILDING SYSTEM GENERATION JOB STREAMS

During the entire system generation process, the user supplies the virtual machine with job streams to perform the individual steps. On a stand-alone system, these job streams are usually kept in card form. Under VM/SP, the CMS editor can be used to create CMS files containing these job streams. Once created, use the VM/SP spooling functions to place these job streams into the virtual machine's card reader whenever they are needed. This technique is especially helpful when the virtual console is some distance from the real reader.

To generate a tape and disk configuration, unload the IBM-supplied base system from a distribution tape to an initialized DASD volume.

Create two small CMS files. These files contain the card reader input data for the two utility programs that occupy the first two files on the distribution tape. The file assumes: (1) the user logs onto VM/SP as user DOSVSY, and (2) the user is in the CMS environment.

The user has entered the following sequence of commands and data entry:

edit initload djcl	(Establish file identification)
input	(Start data entry)
// job intdsk	-----
// assgn syslog,x'01f',c1	↑
// date 12/31/75	
// log	
// assgn syslst,x'00e',l1	
// assgn sysopt,x'250',d4	
// exec	Job
// uid ig	stream
// vtoc strtadr=(0403000),extent=(19)	
vclldossys	
// end	
// job disrst	
// assgn sys005,x'250',d4	
// assgn sys006,x'180',t2	
// assgn syslst,x'00e',l1	
// exec	↓
(null line)	-----
file	(Terminates data entry)
	(Writes file on disk)

The job stream to initialize the disk and unload the tape is now a CMS file. It is identified by a filename of INITLOAD and a filetype of DJCL.

Similarly, a user can create CMS files containing job streams that:

- Unload the IBM-supplied system to disk without initializing the disk
- Add I/O device specifications to the IBM-supplied supervisor
- Define standard labels
- Perform librarian functions
- Assemble and catalog a new supervisor
- Assemble and catalog additional I/O modules
- Back up the new system on tape or disk

COPYING DISTRIBUTION SYSTEM TO DISK

Assume that two job streams are available as CMS files:

- INITLOAD DJCL -- initialize disk at X'250' and load from tape on X'180'
- SKIPINIT DJCL -- skip first file on tape and restore tape on X'180' to disk at X'250'

When userid DOSVSYs logs onto the VM/SP system, the virtual machine has every I/O device it needs except for a magnetic tape unit. Since this device is only required during the tape-to-disk restore operation, it would be inefficient to tie it up for the entire terminal session. Instead, send the VM/SP system operator a message such as:

```
msg op pls mount dos/vs distribution tape as 180
```

The system operator mounts the appropriate tape reel on some idle tape drive, such as 183, and issues the command:

```
attach 183 to dosvsys as 180
```

The user receives an acknowledgement:

```
TAPE 180 ATTACHED
```

The user is now ready to start system generation. If he is not in the CMS environment, issue the command:

```
ipl cms
```

Spool the punch to the user's virtual reader:

```
cp spool punch to *
```

If the disk is to be initialized, punch out the CMS file:

```
punch initload djcl (noheader)
```

The NOHEADER option eliminates the header record normally punched out to identify a card deck.

After a PUNCH command, the user receives an acknowledgement that the file was spooled to the reader.

If the disk had already been initialized, the user would have punched out the file:

```
LOADONLY DJCL
```

Now that the job stream is in the user's virtual reader, perform an IPL from the tape by issuing the CP command:

```
cp ipl 180
```

When the system goes into a wait state, press the PA1 key (or its equivalent) to place the virtual machine in CP console mode. Issue the CP command READY OOC to ready the virtual reader. Then, issue the CP command BEGIN to return control to the system tape.

The virtual machine is now under the control of the utility programs on tape. The virtual machine should reply to all queries as if on a real machine.

When the distributed DOS/VS has been restored to disk, the user receives a RESTORE COMPLETE and an END OF JOB message; the virtual machine is then placed in a disabled wait state, and the console is put into CP mode. To now perform an IPL from address 250, use the distributed system to perform the rest of the system generation procedures.

READYING THE INTERIM SYSTEM

The IBM-supplied supervisor does not contain any I/O device specifications. Until generating a new supervisor with its particular I/O configuration, the user must include the appropriate ADD and ASSGN commands each time he IPLs DOS/VS.

The following job stream files can be created for use in this and other phases of system generation:

```
IPLADD  DJCL  add and assign I/O devices
STDLABEL DJCL  assign standard labels
DSERV   DJCL  display the directories
```

To start the system, assign standard labels, display the directories, and enter the following sequence of commands:

```
cp spool punch to *
punch ipladd djcl (noheader)
punch stdlabel djcl (noheader)
punch dserv djcl (noheader)
cp ipl 250 clear
```

The last command loads the DOS/VS supervisor from unit 250. When the system goes into a wait state, ready the reader as follows:

(press PA1 key or its equivalent)

then enter:

```
ready 00c
begin
```

As each job ends, press the ENTER or RETURN key (depending on the type of terminal) to start the next job.

ASSEMBLING AND LOADING A NEW SUPERVISOR

The job stream to assemble an installation tailored supervisor is another candidate for a CMS file. In addition to the storage advantage, it is easy to assemble different supervisors.

Example: When planning to assemble a supervisor, enter the supervisor macro assembly job stream in a file called ASMBLSU1 DJCL. To generate another job stream to assemble a different supervisor, IPL CMS and enter:

```
edit asmblsu1 djcl
```

This command accesses the supervisor memo file for a CMS EDIT session. Then, use the CHANGE and INPUT subcommands of EDIT to alter the supervisor macro instructions and the job name. Enter or change the SUPVR macro instruction to include ID=2; this specification identifies the new version of the supervisor as \$\$\$SUP2. When all the changes have been made, file the new version under its own name by issuing:

```
file asmblsu2 djcl
```

The user now has two job streams on file, the original and new supervisor. Use this procedure to generate job streams to assemble any number of supervisors.

To assemble these two supervisors, punch the appropriate job stream files to the virtual card reader. While still in the CMS environment, issue the following sequence of commands:

```
cp spool punch to *
punch ipladd djcl (noheader)
punch stdlabel djcl (noheader)
punch asmblsu1 djcl (noheader)
punch asmblsu2 djcl (noheader)
cp ipl 250 clear
```

At this point, use the CMS editor to check the program listings for errors. There is no need to print out a hard copy. Spool the printer output to the virtual reader, return to the CMS environment, create a CMS file from the reader data, and use the EDIT function to scan the listing.

Before loading DOS/VS, include this CP command in the previous sequence of commands:

```
cp spool printer to *
```

This command directs printer output to the virtual reader. (The punch is already spooled to the reader.)

When processing more than one job, keep the output of each job separate by closing the printer and/or punch at the end of each job. When the EOJ ASMBLSU1 message appears and the virtual machine stops on an ATTN 00C, close both the punch and printer files and also assign them spool filenames and filetypes by issuing the commands:

```
#cp close punch name asmblsu1 deck
#cp close printer name asmblsu1 list
```

After issuing these commands, enter a null line to start the next job. While spool files are automatically closed when issuing an IPL command, the CLOSE command can be used on the last job to name the files.

To return to the CMS environment, enter:

```
#cp ipl cms
```

Use the CMS READCARD command to generate CMS files from the reader files. Each READCARD command converts the first file in the reader to a CMS file with the filename and filetype specified in the READCARD command. That file is then deleted from the reader queue. To determine which file is first in the reader, query the reader with:

```
cp query reader all
```

The response itemizes all the files in the reader in sequence and includes such information as file identification number, printer or punch file, and filename.

If the response to the QUERY command was:

```
... 1502 ... PUN ... ASMBLSU1 DECK
... 1503 ... PRT ... ASMBLSU1 LIST
... 1541 ... PUN ... ASMBLSU2 DECK
... 1542 ... PRT ... ASMBLSU2 LIST
```

and the user then issued:

```
readcard supvsr1 text
```

CMS reads file number 1502 and creates a CMS file with a filename of SUPVSR1 and a filetype of TEXT.

However, to check the listing for errors, issue:

```
cp order reader 1503
```

This command reorders the reader queue to bring the listing file first. Then issue:

```
readcard supvsr listing
edit supvsr listing
```

to create a CMS file containing the assembly listing and to access it for the EDIT function. By using EDIT subcommands, a user can now scan the listing for errors. If there are no errors, create a CMS file, SUPVSR TEXT, from file 1502, which is now the first file in the reader queue. If there are errors, issue:

```
erase supvsr listing
```

to discard the assembly listing, and

```
cp purge reader 1502
```

to remove the corresponding assembly text deck.

Now, use the CMS editor to correct the supervisor macro instructions and pass the job stream over to DOS/VS for another assembly.

When a supervisor text deck is ready for cataloging, insert it into a LINKEDIT DJCL job stream that was created from the CMS file, ZLIB BOOKS (previously punched out). To insert the supervisor text deck, issue the following commands:

```
(from the CMS environment)
edit linkedit djcl (access the job stream)
next (position current line pointer)
delete 2 (remove CATALs and BKEND entries)
locate /include (position current line pointer)
getfile supvsr1 text (insert supervisor text deck)
file (replace updated job stream)
```

The updated LINKEDIT job stream can now be passed to DOS/VS for execution. Either execute this phase interactively as on a real machine or again use the CMS editor to tailor the LINKEDIT job stream; that is, remove any unnecessary jobs and remove the PAUSE statements from the jobs that are to be run.

Note: As an option, the user can link-edit the IBM-supplied library and service routines by using the portion of the LINKEDIT file that follows the new supervisor deck.

ADDING I/O MODULES TO DOS/VIS (DOS/VIS RELEASE 34 AND EARLIER ONLY)

The CMS editor can help assemble and catalog any additional IOCS modules. In the CMS environment, create a job stream file containing all the IOCS macros to be assembled and ensure that each macro statement includes the SEPASMB=YES entry. The following series of commands can be entered:

```
cp spool punch to *
cp purge reader all
punch iocsmods djcl
cp spool punch hold
cp spool printer to *
cp ipl 250 clear
```

These commands send the job stream to the virtual reader, hold the punch, and spool the printer to the reader.

When the assemblies have been completed, reload CMS and examine the listings for any errors by entering this series of commands:

```
#cp ipl cms
cp spool printer off
cp spool punch nohold
readcard iocsmods listing
edit iocsmods listing
.
.   (use EDIT subcommands to scan listing)
.
quit
```

If there are errors, use the CMS editor to correct the macro entries and resubmit the job stream. If there are no errors, enter:

```
print iocsmods listing
```

to print the assembly listings.

Enter the following commands to build a job stream that catalogs the IOCS text decks to the system relocatable library:

```
cp change reader all nohold
readcard iocsmods text
edit catalric djcl
input // jcb catalrlb
input // exec maint
getfile iocsmods text
input /*
input /&
file
```

FINAL HOUSEKEEPING

Once the basic system generation is complete, a number of regularly executed housekeeping procedures can be set up as job streams. Consider job streams to perform these procedures:

- Set new standard labels
- Reallocate library sizes
- Condense libraries
- Set automatic condense limits
- Copy the system residence pack (and private libraries) for backup
- Catalog often-used job control statements and linkage editor control statements to the procedure library

To initially create job streams and pass them to DOS/VS for testing, follow the same procedures as for system generation. After verifying the job streams, use one job stream to catalog the other procedures to the system's procedure library.

Note: When cataloging job control statements to the procedure library, an installation can use the CMS editor.

Sample DOS/VS Directory Entries

The following directory entries represent some batch type virtual machines that can be used to run production jobs under DOS and DOS/VS. The operands specified on the OPTION control statement reflect the requirements of the particular system being used. Disk space can either be dedicated or shared with other systems.

A DOS Virtual Machine

```
USER DOSUSER PASSWORD 256K
ACCOUNT ACCTNO BIN3
IPL 350
OPTION ACCT
    CONSOLE 01F 3215
    SPOOL 00B 2501
    SPOOL 00C 2540 R
    SPOOL 00D 2540 P
    SPOOL 00E 1403
    MDISK 350 3330 101 30 OSDOS1 W
    MDISK 351 3330 1 20 UDISK1 W
```


A DOS/VS Virtual Machine

```
USER DOSVUSER PASSWORD 512K
ACCOUNT ACCTNO BIN4
IPL 350
OPTION ECMODE
  CONSOLE 01F 3215
  SPOOL 012 3505
  SPOOL 013 3525
  SPOOL 002 3211
  LINK VMSYS 190 190 RR
  MDISK 191 3330 11 10 UDISKA WR RPASS WPASS
  MDISK 350 3330 100 50 VOSDOS W
  MDISK 351 3330 21 30 UDISK1 W
```

Accessing DOS/VS

This topic assumes that DOS/VS for use under VM/SP has already been generated and that the system residence volume is available on a real disk or minidisk in read/write status. A user can make the system residence volume available in any one of these ways:

- Define the DOS/VS system residence as a read/write disk in the VM/SP directory entry for the userid running DOS/VS. A typical directory entry might look like the following:

```
MDISK 250 3330 101 50 VDOSYS MR RPASS WPASS
```

- Link to the DOS/VS system residence volume using the LINK command. For example, if the DOS/VS system residence is on the 150 disk in the directory entry for the userid DOSRES, a user could enter:

```
link dosres 150 250 w wpass
```

- Have the VM/SP system operator attach the DOS/VS system residence directly to a userid, for that user's exclusive use. When the operator (or another user with Class B command privileges) attaches the disk to one user, no one else may access the volume.

Note: All of the console logs and command examples in this section assume that a DOS/VS system residence is attached to the virtual machine at virtual address 250.

SHARING THE DOS/VS SYSTEM RESIDENCE VOLUME

If VSAM is used, only one user at a time may access the DOS/VS system residence volume in read/write status. This requirement is necessary because only one user can update the VSAM catalog and the label information cylinder. DOS/VS uses this cylinder to keep a record of volumes and files needed by programs that are running.

However, if VSAM is not used and a user wants to share the DOS/VS system residence volume, set up a read/write core image library for link-editing. Then, share the SYSRES libraries in read-only status with other users. This method is acceptable under one of these two conditions:

- Some action has been taken by the installation to provide individual standard label cylinders for each userid that accesses DOS/VS.

--or--

- Different virtual machines use different partitions. For example, one virtual machine uses BG and F4, and another virtual machine uses F1, F2, and F3.

Another method for sharing DOS/VS is for a user to have his own read/write copy of the system residence volume. Then, this user can share this volume with others in read-only status and share the private libraries with other users. Also, a third alternative is to use VSE environment. Release 1 provides for multiple level storage areas on the system residence volume; while Release 2 provides full read/write sharing of the system residence volume.

STANDARD LABEL CYLINDER

To share DOS/VS among virtual machines when unique partitions are not assigned to each virtual machine, a user must provide a unique standard label cylinder for each such virtual DOS/VS user.

When using DOS/VSE with the VSE/Advanced Functions Program Product (5746-XE8), a user can separate label information areas and use a name to access each area. While users must create these areas on the system residence volume, they don't have to (as in DOS/VS Release 34 and earlier releases):

- Place label information areas at the end of the volume, following the normal standard label cylinder
- Modify the IPL communication routines
- Bypass the \$JOBCTLA procedure

For DOS/VS Release 34 (or earlier), the individual standard label cylinders should be placed at the end of the system residence volume (following the normal standard label cylinder). To support unique standard label cylinders, modify DOS/VS as follows:

- The communication region in each DOS/VS virtual machine must point to the appropriate label cylinder for each user. Do this by modifying the IPL communication routines.
- Bypass the procedure \$JOBCTLA. This procedure updates the communication region pointer to the normal standard label cylinder at the end of each job.
- Bypass the label cylinder reset code in the procedure \$\$BSYSWR. \$\$BSYSWR resets the normal standard label pointer whenever a condense function (CONDS) of the MAINT program is requested. As an alternative, never condense a library in any virtual machine using a "non-standard" label cylinder.

Using Virtual Unit Record Devices

When using DOS/VS in a virtual machine, a user must have the following unit record devices, which are normally defined in the VM/SP directory entry:

- A virtual card reader, from which DOS/VS reads the job input stream.
- A virtual printer, which receives all the SYSLST output generated during DOS/VS operation.
- A virtual card punch, which receives SYSPCH output generated during DOS/VS operation.

Depending upon how DOS/VS was generated, a user may need to determine a virtual device address. For example: If DOS/VS expects a 3211 printer at address 002 and no printer is at this address in the virtual machine configuration, define one with the CP DEFINE command:

```
define 3211 002
```

Note: When using CMS to prepare jobs for a DOS/VS virtual machine, use the virtual card punch to spool jobs to the DOS/VS virtual machine.

Before using DOS/VS, find out (from the programmer at the installation responsible for generating and maintaining DOS/VS) what are the virtual device requirements.

A user can control virtual unit record devices with the CP SPOOL command. However, printed or punched output need not have to be printed or punched. For example: When using a technique for alternating between operating systems, a user can spool the virtual printer to the card reader, as follows:

```
spool printer to *
```

This command reads printed output onto a CMS disk so that it can be examined. Also, use the SPOOL command to change the output spooling class of the virtual machine's spooled printer or punch files.

DEFINING THE OPERATOR'S CONSOLE

During DOS/VS system generation, an address is specified for the operator's console. The user's terminal must also be at this address. Usually, DOS/VS expects the operator's console to be at real address 01F. The device has to be generated as a 3215, 3210, or 1052 console. However, when using DOS/VS in a virtual machine, any terminal type can be used as the virtual operator's console.

To find out the virtual machine's terminal address, enter this command:

```
query console
```

If the response indicates that the terminal is not at 01F but at another address, such as 009 (which is a standard VM/SP console address), enter this command:

```
define 009 as 01f
```

This command allows the terminal to now function as the operator's console for both DOS/VS and CMS.

Preparing Jobs for a DOS/VS Virtual Machine

There are several ways to prepare a job stream for a DOS/VS virtual machine:

- Prepare a deck of punched cards that contains such information as DOS/VS job control statements and input files. Place a CP ID statement at the beginning of this deck to indicate the userid of the DOS/VS virtual machine; for example:

```
ID DOSVUSER
```

Then put the cards in the real system card reader. Based on the userid specified on the ID card, VM/SP directs the spool file to the virtual card reader of the DOS/VS virtual machine, which in this case is being run on the userid DOSVUSER.

- Use CMS to create a disk file containing card images identical to the cards submitted in a real card reader for DOS/VS. Use the CP SPOOL command to spool the virtual card punch to the card reader of the DOS/VS virtual machine and use the CMS PUNCH command to punch the card images. In the CMS environment, issue:

```
spool punch to dosvuser  
punch dosjob jcl (noheader)
```

Use the NOHEADER option of the PUNCH command to suppress punching a CMS READ control card at the beginning of the card deck.

A job stream spooled to DOS/VS by either of these methods remains in the card reader of the DOS/VS virtual machine until the user causes DOS/VS to begin reading the job stream from its card reader.

Spooling the card file can be done before or after initializing DOS/VS or at any time while the system is active.

Before using the virtual punch to punch jobs to a virtual machine, take the precaution of clearing any files or card images that may remain in it from previous jobs. The following commands ensure that the virtual punch does not have any other punch files in it:

```
spool punch nocont  
close punch purge
```

A user should also issue these commands to purge any existing reader files of the virtual machine that run DOS/VS:

```
spool reader nocont  
close reader  
purge reader
```

Of course, do not purge any needed files that may be in the reader. To obtain data about existing reader files before they are purged, enter the command:

```
query reader all
```

Note: If any files are open, this command does not tell the user that they are open.

Loading DOS/VS

This topic describes one method for loading DOS/VS in a virtual machine. It shows how to enter the commands and control statements to IPL DOS/VS and how to ready DOS/VS for input jobs. Following the description of the method is an example of how to IPL DOS/VS from the virtual card reader.

IPL FROM THE CONSOLE

When a user is sure that he has all the virtual unit record devices he needs, that the console is properly defined, and that all required DASD units are attached, he should enter this command to make extended control (EC) mode active for the virtual machine because DOS/VS needs EC mode to page:

```
set ecmode on
```

However, if this option is specified in the virtual machine's directory entry, the user does not have to specify it.

To load DOS/VS into the virtual machine, enter the IPL command:

```
ipl 250
```

After a few seconds, the system enters a wait state. On a real system console, the wait light goes on. On the terminal, a user may want to let a few seconds pass to be sure that the wait state has been entered. To verify that the system is in a wait state, enter CP mode and use the DISPLAY command to display the PSW:

```
display psw
```

If bit 14 is a 1, the system is in a wait state. If not, use the BEGIN command to resume program execution.

After determining that the system is in a wait state, cause an attention interruption by pressing the attention key (or equivalent). The following message asks the user to enter the name of the DOS/VS supervisor:

```
OIO3A SPECIFY SUPERVISOR NAME
```

Entering a null line causes DOS/VS to use the default supervisor, \$\$\$SUP1.

Shortly after entering the supervisor name, the system enters the wait state again. Allow a few seconds to pass to be sure that the wait state has been reached. Then, cause another attention interruption with the attention key.

Next, DOS/VS displays a series of messages that display the status of the IPL procedure, followed by a prompting message:

```
0I10A GIVE IPL CONTROL COMMANDS
```

In response to this message, enter these commands in this order:

1. The ADD or DEL commands to optionally alter the default DOS/VS configuration, which is established at system generation.
2. The SET command (required) to initialize the date and time clock.
3. The CAT command to optionally define the VSAM master catalog.
4. The DPD command (required) to define the page data set.

After entering the DPD command, this message appears:

```
0I20I DOS/VS IPL COMPLETE
```

It signifies that DOS/VS is loaded into the virtual machine.

If a warm start copy of the SVA (shared virtual area) is available, this message also appears:

```
1I00A WARM START COPY OF SVA FOUND
```

In response, enter KEEP (or a null line) or REJ, depending upon whether this copy of the SVA is to be used.

If no warm start copy of the SVA is available and the SVA must be used, create one by using a standard procedure, depending upon what is available in DOS/VS.

When the IPL procedure is complete, this message appears:

```
BG 1I00A READY FOR COMMUNICATIONS
```

It indicates that the background partition is running and is ready to accept control commands or job control statements.

The complete VM/SP logon and DOS/VS IPL procedure as it would appear on a 3270 terminal is shown in Figure 15. The exclamation marks (!) indicate pressing the attention key (or its equivalent).

```

| logon tester
| ENTER PASSWORD:
|
| FILES: 001 RDR, NO PRT, NO PUN          (see Note 1)
| LOGON AT 08:19:52 EST MONDAY 01/12/76
| set ecmode on
| link dcsys 250 250 w          (see Note 2)
| DASD 250 LINKED R/W
| ipl 250          (see Note 3)
| !
| OI03A SPECIFY SUPERVISOR NAME
| $$a$sup2
| !
| OI04I IPLDEV=X'250',VOLSER=DOSRE3,CPUID=PF0101530155
| OI30I DATE=01/12/76,CLOCK=16/35/09,ZONE=EAST/04/00
| OI10A GIVE IPL CONTROL COMMANDS
| set          (see Note 4)
| dpd
| OI52I PAGE DATA SET EXTENT          LOW          HIGH
|                                     327          0 250 11
| OI20I DOS/VS IPL COMPLETE
| BG
| 1I00A READY FOR COMMUNICATIONS.
| BG          (see Note 5)
| log
| BG
| assgn sysrec,x'250'
| BG
| set rf=yes
| BG
|          (see Note 6)
| BG
| // JOB TEST1
|
| DATE 10/30/75,CLOCK 16/35/47
| BG
| 1I89A IPL REASON CODE =          (see Note 7)
| BG
| 1I91A SUB-SYSTEM ID =
| BG
| 1I93I RECORDER FILE IS 2% FULL
| BG
| // OPTION NODECK          (see Note 8)
| BG
| // EXEC ASSEMBLY
| BG
| EOJ TEST1
|
| DATE 10/30/75,CLOCK 16/37/49,DURATION 00/02/01
| BG
| 1C00A ATTN. 00C          (see Note 9)
| BG
|          (see Note 10)
| BG
| OP08A INTERV REQ SYSRDR=00C

```

Figure 15. IPL DOS/VS and Execute a Job in the Card Reader (Part 1 of 2)

Notes

1. The reader file contains a job stream for the DOS/VS virtual machine.
2. The DOS/VS system residence is assumed to be defined at virtual address 250 in userid DOSYS's directory entry and have a write password of ALL.
3. After the IPL command, the attention interruption causes message 0I03A to be issued. After entering the supervisor name, a later interruption continues the IPL procedure.
4. The SET and DPD commands are required to IPL DOS/VS.
5. The background partition is active and waiting for commands. The LOG command is optional; the ASSGN and SET commands shown here may be required for system recording.
6. A null line signals an interruption to the card reader, so that DOS/VS begins reading the job stream.
7. These are RDE (reliability data extractor) messages; a null line entered in response indicates that the user is taking the default values.
8. DOS/VS continues reading the job.
9. The card reader is empty; the spool file has been read.
10. A null line causes another interrupt to the card reader; if another file is in the card reader, it is read. Otherwise, message 0P08A is issued.

Figure 15. IPL DOS/VS and Execute a Job in the Card Reader (Part 2 of 2)

IPL FROM THE CARD READER

Place the control commands for performing the IPL procedure in the card reader of the DOS/VS virtual machine before issuing the IPL command to load the system. Then, signal DOS/VS to read the control commands from the card reader. This method can be more efficient than entering all of the control commands manually, especially if there are many ADD and DEL commands that must be issued when initializing DOS/VS.

The cards required to perform the IPL procedure must be in two separate card files: (1) a single card specifying the supervisor name, and (2) a card deck containing the IPL commands. A user may follow these card files with additional files that contain jobs for execution in the DOS/VS virtual machine.

To load DOS/VS into the virtual machine, enter the IPL command:

```
ipl 250
```


When the system enters a wait state, IPL DOS/VS from the card reader by causing a reader interruption. This interruption allows the system to read the name of the supervisor from the card reader. To cause a reader interruption, enter the CP environment by pressing the attention key twice (or the 3270's PA1 key once) to cause a CP read, and enter these commands:

```
#cp ready 00c
#cp begin
```

The READY command causes a reader interruption; the BEGIN command returns control to DOS/VS which reads the first spool file from the card reader.

Shortly after this card file is read, the system enters the wait state a second time. The user must again enter the CP environment and enter these commands:

```
ready 00c
begin
```

In response to these commands, DOS/VS begins reading the IPL commands.

Note: When initializing DOS/VS through the card reader, a user cannot supply the responses necessary to save the warm start copy of the SVA from card input. These responses must be supplied from the console. However, to create the SVA and SDL during IPL, place the cards to perform these procedures in the card reader in a separate spool file following the IPL commands. Again, the user must enter the READY 00C command to cause the reader interruption to force the system to begin reading from the card reader.

DOS/VS Operation

Depending upon how DOS/VS was generated, there may be additional operator commands and control statements that must be entered at the console running jobs on the DOS/VS virtual machine.

If there is an active system recorder file, it is opened when the first // JOB card is encountered in the input stream. Before this JOB card is read, enter the ASSGN and SET commands that define the SYSREC device and the status of the system recorder file. Otherwise, an error is encountered opening the SYSREC file. For example: If the system recorder file is already active on the system residence volume, enter:

```
assgn sysrec,X'250' (for DOS/VS Release 34 or earlier only)
set rf=yes
```

When starting a DOS/VS virtual machine to run in batch mode to process jobs from other users, a user may also want to enter the operator commands necessary to:

- Allocate storage among different partitions
- Start foreground partitions in operation
- Initialize POWER/VS

These considerations are discussed later in this section under the topic "Running Batch DOS/VS under VM/SP." When alternating between CMS and DOS/VS, a user may want to keep the IPL procedure as simple as possible.

STARTING A JOB STREAM

After preparing job streams and placing them in the DOS/VS card reader, the message READY FOR COMMUNICATIONS appears. Enter a null line (with the return or enter key) to cause an interruption. This interruption causes DOS/VS to begin reading from the card reader.

If the LOG ccommand was entered before beginning the job stream, DOS/VS displays on the user's terminal all the job control statements that are executed.

As the DOS/VS operator, the virtual machine user can enter control statements directly from the terminal. For example: To execute cataloged programs or procedures, enter the ASSGN, DLBL, and EXEC statements necessary to execute the program directly from the console.

WHEN A JOB IS FINISHED

When the DOS/VS virtual machine running a single partition finishes reading a spool file (which may contain one or more jobs), it posts this attention message to indicate that the reader is empty:

```
1C00A  ATTN. 00C
```

If additional jobs are in the card reader and were spooled as separate CP spool files, enter a null line to cause DOS/VS to begin reading from the card reader. If no more files are in the reader and the user enters a null line, this message appears to indicate that operator intervention is required:

```
0P08A  INTERV REQ SYSRDR=00C
```

If desired, users can put cards into the real system card reader to direct another job stream to the DOS/VS virtual machine. When the DOS/VS virtual machine receives cards in its reader, it begins reading them automatically.

If the card reader of the DOS/VS virtual machine is spooled with the CONT operand, then any jobs received in the card reader while a job is executing are read upon completion of the current job stream. If a job stream completes and if the end of the spool file is reached before another job is received, an interruption must be issued to cause the next job to be read.

To terminate the DOS/VS terminal session, use the attention key (or equivalent) to enter the CP command environment. Under CP, either enter the LOGOFF ccommand to log off the VM/SP system or use the IPL command to load another operating system into the virtual machine.

COMMUNICATING WITH CP

While operating the DOS/VS virtual machine, use CP commands to:

- Communicate with the system operator or other virtual machine users
- Query the status of virtual machine devices or spool files
- Attach or detach devices from the virtual machine configuration

To enter a CP command while the DOS/VS virtual machine is in operation, press the attention key (or its equivalent) twice, quickly, to force a CP read. After entering the necessary CP commands, use the CP BEGIN command to resume DOS/VS virtual machine execution. For example: If a DOS/VS PAUSE control statement requests the VM/SP operator to perform some action, the user must enter the CP environment to send a message to the VM/SP system operator. The following lines represent a typical sequence on a typewriter terminal:

```
// PAUSE ASK OPERATOR TO ATTACH TAPE AS 284
!!
CP
msg op please attach scratch tape as 284
sleep
TAPE 284 ATTACHED
!
CP
begin
```

The SLEEP command locks the keyboard and places the virtual machine in a dormant status to wait for any messages. When the message appears indicating the tape is attached, press the attention key once to return to CP, and then issue the BEGIN command to resume virtual machine execution. Because the // PAUSE message is still outstanding, press the RETURN key to continue DOS/VS execution.

Using the #CP Function

In most cases during virtual machine operation, use the #CP function to enter CP commands directly from the virtual machine environment. Users do not always have to press the attention key (or its equivalent) to enter the CP environment. For example: If the virtual machine is waiting for a read, a user can enter a CP command with the #CP function:

```
#cp msg op please mount vol240
```

The command line is processed by VM/SP, and the virtual machine is still waiting for a read. This method may be convenient for entering CP commands.

Note: At times CP commands cannot be entered with the #CP function. For example: During the IPL procedure when DOS/VS is processing IPL commands, the CCWs used for these reads expect only three bytes of data. Any additional information on CP command lines is truncated.

Interrupting the Virtual Machine

While DOS/VS is running in the virtual machine, a user can interrupt its execution by using the attention key (or its equivalent) on the terminal. When this key is pressed, the DOS/VS attention handler responds with these messages:

```
AR
1I60A  READY FOR COMMUNICATIONS.
AR
```

When these messages appear, a user can enter attention commands. To resume program execution, enter a null line. Normally, wait until the attention has been processed before signaling another one. An exception would be when canceling a dump.

When using a 2741 terminal and its attention key mainly to signal CP interruptions, enter the command:

```
#cp terminal mode cp
```

The first time the attention key is pressed, VM/SP posts a CP interruption. The next pressing of the attention key signals an interruption to DOS/VS. When a user is in the CP environment and wants to signal an interruption to DOS/VS, enter either the ATTN or REQUEST commands.

Running Batch DOS/VS Under VM/SP

When using DOS/VS in a virtual machine as a production tool, it is likely that the virtual machine running DOS/VS is going to be logged on continuously. This machine may be available for many users to submit jobs, or it may be used only by installation personnel responsible for running the production jobs.

In either event, it is likely that DOS/VS has been generated specifically for use under VM/SP. In this case, a user should know whether:

- It is necessary to start more than one partition. If it is, determine how much virtual storage to allocate to each partition. These considerations are much the same as they would be for a native DOS/VS user, who must decide how much work is going to be done and what is the most efficient way to do it.
- To generate the POWER program (POWER/VS for DOS/VS or VSE/POWER in the VSE environment). This program can provide many functions that are not available when relying solely on CP spooling, such as spooling jobs via class and purging selected jobs in a job stream.

When the POWER program is active in the DOS/VS virtual machine, it controls which jobs are to be processed in the various partitions, and it reads jobs from the card reader. Because the program is constantly working, no need exists for an attention interruption when a new job is placed in a card reader.

To start the POWER program in a virtual machine, use the AUTOSTART procedure. Enter the card deck for the automatic start either using cards in the reader or through the virtual card reader.

- Virtual devices are dedicated for use by the DOS/VS virtual machine or devices must be shared among other virtual machines.

For example: If a card reader has been dedicated to a DOS/VS virtual machine, then users may submit jobs through that card reader. They do not have to place a CP ID card at the beginning of the deck to direct it to the appropriate virtual machine. If the DOS/VS virtual machine is sharing the system card reader, then any user submitting a card deck must place a CP ID card at the beginning of the deck, specifying the userid of the DOS/VS virtual machine.

Alternating between CMS and DOS/VS Under VM/SP

When working in a development and testing environment (rather than in a batch environment) and the work is not suitable for CMS/DOS, these are some advantages to alternating between CMS and DOS/VS:

- Reduced unit record output. Under CMS, users can examine program output and compiler listings online, check the results and resubmit the job without printing a single page on the system printer or punching card decks.
- Faster turnaround time compared with batch alone; under CMS, users can see the results of program compilation and execution immediately, rather than waiting for output from a batch system.

This topic outlines the technique for alternating between operating systems. Before using this technique, users should be familiar with CMS, the conversational component of VM/SP, particularly with the CMS editor and some file system commands. CMS usage information is in VM/SP CMS User's Guide. For details on CMS commands and EDIT subcommands, refer to VM/SP CMS Command and Macro Reference.

LOADING CMS INTO A VIRTUAL MACHINE

To load CMS into a virtual machine, use the IPL command. Usually, IPL CMS by specifying the saved system name CMS:

```
ipl cms
```

Or, load CMS by specifying the virtual address of the CMS system, which is usually 190:

```
ipl 190
```

After receiving a message like this:

```
CMS VERSION 3.0
```

A user can use the interactive facilities of CMS to prepare jobs for execution in the DOS/VS virtual machine.

USING THE CMS EDITOR TO PREPARE JOBS

Use the EDIT command in CMS to pass control to the CMS editor for preparing disk files of 80-character card image records. The created files may contain DOS/VS job control statements, source files, or even IPL decks.

For example: To IPL DOS/VS using input statements from the card reader, prepare CMS files that contain:

- The DOS/VS supervisor name required for an IPL, such as the record:

```
$$$SUP1
```

- The IPL control statements that a user wants to supply, such as:

```
set
dpd
assgn sysrec,x'250'
set rf=yes
log
```

Assume for the purposes of this example that: (1) these files have CMS file identifications of SUPER JCL and START JCL, and (2) in these files are all the statements needed to control DOS/VS IPL.

The CMS editor can also be used to prepare the job stream selected for execution. For example: To assemble an assembler language source program and make the source file available as a CMS disk file, edit the file and insert the DOS/VS job control statements into the CMS file. The file DOSTEST JCL might contain:

```
// JOB TEST1
// OPTION NODECK
// EXEC ASSEMBLY
  (assembler language source statements)
  .
  .
  .
/*
/8
```

Although this job stream contains only the job control statements necessary to assemble the job, it can also include the statements necessary to link-edit the output module, catalog the program, and so on.

ISSUING SPOOL COMMANDS TO CONTROL UNIT RECORD DEVICES

Before sending a job to the DOS/VS virtual machine, check the unit record devices:

- Close and purge the card reader, to make sure no files are left over from a previous job:

```
spool reader nocont
close reader
purge reader all
```

- Close and purge the card punch, and spool it to the card reader:

```
spool punch nocont
close punch purge
spool punch to *
```

When using the card reader to IPL DOS/VS, spool the punch NOCONT. At least the first two files must be separate spool files.

- Spool the virtual printer to the card reader:

```
spool printer to *
```

For DOS/VS SYSST output to print on the system printer, omit this SPOOL command. A user can also spool the printer file to a different class, such as L. This action ensures that the printer is closed before the user is ready to IPL CMS and that DOS/VS does not read the printer file from the reader:

```
spool printer to * class l
```

PUNCHING CMS FILES

Use the CMS PUNCH command to punch the card files. The files are spooled to the virtual card reader. For example, to punch the three files used above, enter:

```
punch super jcl (noheader
punch start jcl (noheader
punch dostest jcl (noheader
```

The NOHEADER option must be used to suppress punching a CMS READ control card, which is punched by default when using the PUNCH command.

INITIALIZING DOS/VS

When the DOS/VS system residence volume is attached to the virtual machine, use the IPL command to load DOS/VS:

```
ipl 250
```

After waiting a few moments, use the DISPLAY command to see if the system is in a wait state. After determining that the system is in a wait state, enter an attention interruption:

```
!!
CP
display psw
PSW = 030EC0C 00000000
ready 00c
begin
```

After this BEGIN command returns control to the virtual machine, the file SUPER JCL is read from the card reader, and the IPL procedure continues. Again, the user must wait for the system to enter a wait state and ready the reader:

```
!!
CP
display psw
PSW = 030E0000 00012800
ready 00c
begin
```

When the IPL procedure is complete, such messages as these appear:

```
0I52I PAGE DATA SET EXTENT      LOW      HIGH
                                327  0    250  11
0I20I IPL COMPLETE FOR DOS/VS REL xx.x ECLEVEL=nn
BG
1C00A  ATTN. 00C
BG
```

As a practical matter, an installation may want to create a DOS/VS saved system at this point so that the future loading of DOS/VS would avoid these preliminary steps. Refer to the VM/SP System Programmer's Guide for information on creating saved systems.

SIGNALING DOS/VS TO READ THE JOB STREAM

When DOS/VS has been loaded and the message BG indicates that it is waiting for communication, enter a null line. This action causes DOS/VS to begin reading the next spool reader file, which is the one that contains the job control statements, and in this example, the assembler language source program.

If the LOG command is issued during the IPL procedure, all of the job control statements that are read are displayed on the terminal as they are executed.

Note: If the RDE option is in effect for DOS/VS Release 34 (or earlier), respond to these messages before the job is executed:

```
1I89A  IPL REASON CODE =
1I91A  SUB-SYSTEM ID =
```

When the Card Reader Is Empty

When the job sent from the punch to the card reader has been read, this message appears, followed by a time stamp:

```
EOJ TEST1
```

The time stamp could look like this:

```
DATE 02/19/76,CLOCK 20/19/28 DURATION 00/07/48
```

Additional messages from the background partition can also appear, such as:

```
BG
1C00A  ATTN. 00C
BG
```

When punching more than one job and the spool files for remaining jobs are still in the card reader, enter another null line to signal DOS/VS to read from the reader.

RELOADING CMS INTO A VIRTUAL MACHINE

When a job or jobs are finished in the DOS/VS virtual machine, a user may return to the CMS environment by entering CP mode and issuing this command:

```
ipl cms
```

This IPL command closes the virtual printer. If the printer is spooled to the card reader or if the printer is in a hold status, a message from VM/SP appears, such as:

```
PRT FILE 4786 TO BILBO COPY 01 NOHOLD
```

```
--or--
```

```
PRT FILE 4786 FOR BILBO COPY 01 HOLD
```

This printer file contains the SYSLST output from the job that executed in DOS/VS.

If the file is spooled to the printer, it is queued for printing on the system printer. If the printer is spooled to the card reader, it is a reader file that is available to the user in the card reader.

EXAMINING DOS/VS VIRTUAL MACHINE OUTPUT

To examine a job's output executed under the DOS/VS virtual machine, spool the printer to the card reader and read the reader file onto the CMS A-disk by using the READCARD command:

```
readcard dostest output
```

This command creates a file named DOSTEST OUTPUT, which contains the spooled printer output. Now, using the CMS editor, either examine the contents of the file or use the TYPE command to display the whole file at the terminal.

Use the EDIT command to pass control to the CMS editor:

```
edit dostest output
```

To examine assembly or compilation output, use a LOCATE subcommand to locate a keyword in the listing file. For example: To locate the beginning of the diagnostic messages produced by the assembler, enter the subcommand:

```
locate /diagnostics
```

If no errors are in the assembly, modify the job control statements in the file and resubmit the job. Otherwise, correct the source statements and then start over. Starting over means that a user must repeat the procedure for clearing files from the card punch and card reader, punching the IPL decks and the job stream, and reinitializing DOS/VS.

USING EXEC PROCEDURES

When alternating extensively between CMS and DOS/VS in a virtual machine, place in an EXEC procedure the commands necessary to spool unit record devices, punch the IPL deck, and punch a job stream. Then, when sending a job to the card reader and initializing DOS/VS, a user only has to enter the EXEC name and, in some cases, a few additional control commands.

For example: The command sequence (previously mentioned) could be contained in an EXEC procedure, such as:

```
CP SPOOL PUNCH NOCONT
CP CLOSE PUNCH PURGE
CP CLOSE C
CP PURGE READER ALL
PUNCH SUPER JCL (NOH
PUNCH START JCL (NOH
PUNCH DOSTEST JCL (NOH
CP IPL 250
```

If this EXEC procedure is named DOSJOB, then to execute this procedure, all a user has to enter is:

```
dosjob
```

Once the IPL command in the EXEC is performed, the virtual machine is no longer under the control of CMS. To begin DOS/VS operation, enter attention interruptions and CP commands, as necessary.

A user can make an EXEC procedure more generalized. For example: To make this EXEC capable of sending any job to the card reader, use the EXEC symbol &1 in place of the DOSTEST filename, such as:

```
CP SPOOL PUNCH NOCONT
CP CLOSE PUNCH PURGE
CP CLOSE C
CP PURGE READER ALL
PUNCH SUPER JCL (NOH
PUNCH START JCL (NOH
PUNCH &1 JCL (NOH
CP IPL 250
```

With this EXEC, a user can send any job to the card reader that has a CMS filetype of JCL. For example: If a user enters:

```
dosjob prog3
```

The EXEC variable symbol &1 is replaced with the argument PROG3 and the file PROG3 JCL is punched to the card reader for execution under DOS/VS.

As an additional aid when entering the commands and responses necessary to IPL DOS/VS from the card reader, use the CONT option of the SPOOL command. This option allows a user to spool the IPL commands and job stream as a single spool file. For example:

```
CF SPOOL PUNCH NOCONT
CP CLOSE PUNCH PURGE
CF CLOSE C
CP PURGE READER ALL
PUNCH SUPER JCL (NOH
CP SPOOL D CONT
PUNCH START JCL (NOH
PUNCH DOSTEST JCL (NOH
CP SPOOL D NOCONT
CF IPL 250
```

When spooling the punch with the CONT operand, VM/SP spools the START JCL and DOSTEST JCL files as a single spool file. When the IPL commands are read, DOS/VS continues reading from the card reader. A user does not have to signal (with a null line) when DOS/VS is to begin reading the job stream.

Continuous spooling has an additional advantage in the IPL procedure. When issuing the IPL command, the virtual punch is closed. However, in the previous example where the punch was spooled NOCONT, but not closed, the IPL closes the punch. As the file is spooled to the reader, this action causes a reader interruption, which is the first interruption that a user has to enter (the one that causes DOS/VS to read the supervisor name).

Thus, when executing this EXEC format, a user is required (after the IPL command is executed) to enter only one interruption and one READY command to execute the entire job. The console sheet might look like this:

```
dcsjob
!!
display psw
PSW = 030E0000 000128000
ready 00c
begin
```

After issuing this BEGIN command, the remainder of the job is executed without user intervention until the card reader is empty.

The CMS EXEC facility is described in detail in the VM/SP CMS User's Guide.

Using More Than One Virtual Machine

In a DOS/VS virtual machine, if a user is running a job that may take a long time to execute, he may want to free his terminal for other work. In these situations, use the DISCONN command to disconnect the DOS/VS virtual machine and logon to some other userid.

This topic shows a sample procedure (using the userids CMSPREP1 and DOSTEST1), followed by a list of some additional things that must be considered when disconnecting a terminal.

Note: While it is not necessary to use CMS to prepare or transmit the jobs to the DOS/VS virtual machine, users may find it convenient.

ACCESSING CMS

To access CMS, a user must first have a userid. A user can logon to that userid and load CMS as follows:

```
logon cmsprep1
ENTER PASSWORD:

LOGON AT 10:30:41 EST TUESDAY 02/28/76
ipl cms
CMS VERSION 3.0
```

Assuming that the user has a read/write CMS A-disk, a user can create CMS files that contain the IPL control commands and responses, as well as the job streams to execute in DOS/VS. Then, by using the SPOOL and PUNCH commands, place copies of these files in the card reader of the machine that is going to run DOS/VS.

For example: A user has two files named SUPER JCL and START JCL, which contain the supervisor name and IPL commands for the IPL procedure. In addition, the file DOSTEST JCL contains the job stream that a user wants to execute. To spool these files to the userid DOSTEST1, enter:

```
cp spool punch to dostest1
punch super jcl (noheader
punch start jcl (noheader
punch dostest jcl (noheader
```

DISCONNECTING CMS

When a user is ready to logon to the userid that is going to run DOS/VS, disconnect the CMS virtual machine as follows:

```
disconn hold
```

A user should use the HOLD operand of the DISCONN command when he uses a dial-up terminal and does not want to lose the connection.

Since the CMS machine is not currently active, there is no need to disconnect it. While a user could just as easily log off, disconnecting and logging on again saves reloading CMS or respooling the punch, printer, and so on.

LOGGING ONTO DOS/VS

When logging onto the virtual machine that is going to run DOS/VS, a user sees, in addition to the normal log messages, a message indicating that there are files in the card reader:

```
logon dostest1
.
FILES: 003 RDR, NO PRT, NO PUN
LOGON AT 10:50:34 EST TUESDAY 02/28/76
```

If this virtual machine does not have the ECMODE option set on (determined by issuing the QUERY SET command), issue the command SET ECMODE ON. Then, begin the IPL procedure as outlined under the topic "IPL from the Card Reader" (described previously in this section).

Note: If the RDE option is in effect for DOS/VS Release 34 (or earlier), wait until after responding to the RDE messages before disconnecting the DOS/VS virtual machine. Then, enter CP mode, using the attention key, and enter the SET RUN ON and DISCONN commands:

```
!!
CP
set run on
disconn
```

The DOS/VS virtual machine continues to run.

RETURNING TO CMS

A user can now use the VM/SP logon procedures to reconnect to the CMS virtual machine. During logon, this message appears instead of the normal LOGON message:

```
RECONNECT AT 11:05:02 EST TUESDAY 02/28/76
```

To return to the CMS environment and continue working, enter the BEGIN command.

To see if the virtual machine that is set up to run DOS/VS is running, issue this command:

```
query dostest1
```

After issuing this command, a user can continue to use CMS (or some other operating system) in this virtual machine.

If desired, users can again disconnect the CMS virtual machine and reconnect onto the DOS/VS virtual machine. For example: A user knows that the job stream being executed may pause to require an operator response. Or, a user may use CMS to spool another job to DOS/VS. Then, a user may need to reconnect to the DOS/VS virtual machine to alert DOS/VS to begin reading from the card reader.

After reconnecting, issue the BEGIN command for the DOS/VS virtual machine to resume execution.

Note: By using the single console image facility described in the VM/SP System Programmer's Guide, both the CMS virtual machine and the DOS/VS virtual machine can run concurrently from the same terminal. There is no need for repeated disconnecting and reconnecting of these virtual machines.

DISCONNECTION CONSIDERATIONS

When using more than one userid to alternate between two operating systems, consider:

- How DOS/VS may read additional jobs from the card reader?
- What happens when there is a read from the console of the DOS/VS virtual machine?
- What happens to the console log of a disconnected virtual machine?
- Could the single console image facility be useful in controlling multiple virtual machines concurrently from one terminal?

Sending Jobs to a Disconnected DOS/VS Machine

When running DOS/VS disconnected, a user must reconnect to the DOS/VS virtual machine to enter the interruption that causes DOS/VS to continue reading jobs. However, a user can avoid this annoyance by using the single console image facility. With this facility, a designated user can generate the interruption with the CP SEND command; in this case he does not have to reconnect the DOS/VS virtual machine.

To use CMS to spool jobs to a disconnected DOS/VS virtual machine, spool the reader of the DOS/VS machine with the CONT operand:

```
spool reader cont
```

Then, if another job is received in the card reader of the DOS/VS machine before the currently executing job stream has completed, the job just received is also read and executed. Thus, when running a series of jobs in a single job stream and the reader is spooled with the CONT operand, a user can continue to send (from the CMS virtual machine) additional jobs to the DOS/VS virtual machine for execution.

Note: Under these circumstances, a user does not have to reconnect to the DOS/VS virtual machine to signal an interruption, unless the currently executing job stream finishes before the next job is received in the card reader.

If DOS/VS is running disconnected and is processing many jobs, a user may want to obtain printed output for completed jobs without waiting for all the jobs to finish.

When running DOS/VSE with the VSE/Advanced Functions Program Product (5746-XE8), VM/SP automatically releases spooled printer output to VSE/POWER. However, for DOS/VS Release 34 (or earlier), release spooled printer output (either accumulated directly or through the POWER/VS program) by reconnecting to DOS/VS and issuing the command:

```
close printer
```

This command releases the SYSLST output accumulated thus far. However, when using the POWER/VS program and a dedicated printer, all spooled output files are under the control of POWER/VS spooling, and not VM/SP spooling. Also, issuing the CLOSE command from a secondary user's console avoids the necessity to reconnect the DOS/VS virtual machine in order to release output.

Console READS and WRITES in a Disconnected DOS/VS Machine

- Without the single console image facility:

When running DOS/VS disconnected, a 15-minute time-out begins when a console read occurs. If the virtual machine does not respond to the read before the 15 minutes elapse, VM/SP automatically logs off the virtual machine.

For example: DOS/VS is running disconnected, and a program running in the machine completes execution or issues a PAUSE request. The virtual machine is logged off after 15 minutes unless the user reconnects the virtual machine and issues the BEGIN command.

When running DOS/VS disconnected, VM/SP ignores all output or "writes" to the virtual console unless the console is spooled. To spool virtual console output, a virtual machine user can issue the following CP command before disconnecting the virtual machine:

```
#cp spool console start
```

This command starts recording all console output on a spool file. When the user logs on again, he can issue:

```
spool console stop close
```

This command stops console spooling and releases the spool file to the real printer. If this command is not entered, VM/SP saves all spooled console output except the last 4K page of output.

- With the single console image facility:

If a disconnected virtual machine with an active secondary user issues a read to the console, a message is sent to the console informing the secondary user. No 15-minute time-out is initiated. The secondary user then satisfies the read by issuing a SEND command, containing the appropriate information, to the disconnected virtual machine.

Any console output from a disconnected virtual machine with an active secondary user will appear on the console of the secondary user. Each output line will have a prefix consisting of the disconnected virtual machine's userid followed by a colon. Also, console spooling can be used as described in "Without the single console image facility".

Developing and Testing Programs to Run in a DOS/VS Virtual Machine

The foregoing discussions noted how to use the CMS editor and the EXEC facility to help prepare jobs for execution in a DOS/VS virtual machine. In addition to these CMS features, there are a number of other CMS commands for developing and testing programs on CMS disks. One advantage of storing source programs on CMS disks is that they can be maintained as backup copies of a program while a second version is being tested and debugged.

CMS has a special environment, called CMS/DOS, which provides many commands that simulate the functions of DOS/VS or the VSE environment. An installation can use VM/SP to support FB-512 devices in its configuration. (In order to access FB-512 devices in the VSE environment, an installation also needs to install the VSE/VSAM Program Product, Program No. 5746-AM2.) Without VM/SP, the VM/370 Basic System Extensicns Program Product (5748-XX8) must be installed in order to use FB-512 devices.

Some of the things that can be done in CMS/DOS are:

- Creating CMS macro libraries from DOS/VS macro libraries and assembling programs directly in CMS. (Assembler diagnostic messages are displayed on the terminal.)
- Compiling programs written in DOS/VS COBOL or DOS/VS PL/I programming languages, using DOS/VS macro libraries.
- Displaying or printing the directories of DOS/VS private or system core image, relocatable, source statement, or procedure libraries.
- Displaying or printing the procedure library.
- Link-editing TEXT decks from CMS disks, or relocatable modules from DOS/VS or private relocatable libraries. These simulated core image libraries called DOSLIBs are on a CMS disk. Users can also copy relocatable modules from DOS/VS libraries.
- Loading core image phases from CMS DOSLIBs or from DOS/VS core image libraries into virtual storage and executing them.
- Identifying system and programmer logical units for programs being used and listing current assignments.
- Identifying disk files. When executing programs in CMS/DOS, users can read sequential disk files directly from DOS disks, but cannot write on them. Instead, for testing purposes, write CMS disk files or output files to the virtual punch or printer, or to a tape. An exception to this rule is when executing COBOL and PL/I programs in CMS/DOS. CMS can be used to both read and write VSAM files located on DOS disks.
- Using the CMS and VM/SP debugging facilities to debug a program under CMS. Users can set address stops (called breakpoints in the CMS debug environment) and temporarily halt the execution of a program to examine or change the contents of registers or specific storage locations.

When a program is tested and debugged in CMS/DOS, users can also prepare a job stream to catalog and execute the program in a DOS/VS virtual machine.

For complete details about how to use CMS/DOS, refer to VM/SP CMS User's Guide. For details about how to specify CMS commands, refer to VM/SP CMS Command and Macro Reference.

Summary

When a virtual machine user loads DOS/VS into his virtual machine, the terminal becomes the DOS/VS operator console, and the virtual machine user becomes the operator responsible for entering all commands and responses. The three basic techniques for using DOS/VS in a virtual machine are: running DOS/VS in batch mode, using the IPL command to alternate between DOS/VS and CMS in a single virtual machine, and running DOS/VS disconnected.

Before using one of these techniques, an installation must understand how to:

- Generate DOS/VS to run in a virtual machine
- Create VM/SP directory entries for DOS/VS virtual machines
- Access the DOS/VS system residence volume
- Ensure that the proper I/O devices are attached to the DOS/VS virtual machine
- IPL and operate DOS/VS under VM/SP

The primary objectives when generating DOS/VS to run in a virtual machine should be to avoid double CCW translation and to reduce the number of SIO instructions issued by DOS/VS. To meet these objectives, an installation needs to consider how it generates both VM/SP and DOS/VS. (DOS/VS can be generated under VM/SP.)

DOS/VS operation depends upon how DOS/VS was generated. There may be additional operator commands and control statements that must be entered at the console before running jobs on the DOS/VS virtual machine.

There are many ways that DOS/VS virtual machine users can be helped by using the CMS component of VM/SP. The CMS editor and EXEC facility can be used to help prepare jobs for execution in a DOS/VS virtual machine. CMS commands can be used to develop and test programs on CMS disks. The CMS/DOS environment provides many commands that simulate DOS/VS functions.

Section 4. OS/VS in a Virtual Machine

This section uses the term "OS/VS" as a generic expression. It represents any or all of the OS, OS/VS1, OS/VS2 SVS, and OS/VS2 MVS system control programs unless specified otherwise.

When loading OS/VS into a virtual machine running under VM/SP, the terminal becomes the OS/VS operator console, and the user is responsible for entering all the commands and responses normally required of the operator.

The four basic techniques to use when running OS/VS in a virtual machine are:

- Batch mode. One user runs as the OS/VS machine (userid OSVS) and other users (like CMSID1) may submit jobs either through the virtual card reader, through the system card reader, or via JES remote stations.
- Alternating technique. The IPL command is used to alternate between OS/VS and CMS in a single virtual machine. This method requires only one directory entry, the one for the OS/VS user. Because of the lengthy IPL process, it is practical only if an installation has created an OS/VS saved system.
- Disconnected user (OS/VS2 only). Because OS/VS2 allows a user to recall a prior system message, he can logon as an OS/VS2 operator under the OSVS userid, start up his system, and then disconnect. While the OS/VS2 machine continues to run, the user can logon as a CMS user (CMSID1) and create and submit job streams and check the resulting output. To check the progress of the operating system, disconnect from the CMS machine and reconnect to the OSVS virtual machine via the LOGON command.
- Disconnected, however, the CMS user who is using the single console image facility acts as the secondary user for the OS/VS user. This allows the CMS user to control messages, replies, and commands for the disconnected OS/VS user at the same physical terminal. The CMS user can perform this function without reconnecting the OS/VS virtual machine.

Before discussing these four techniques in greater detail, the user must understand how to:

- Generate OS/VS to run in a virtual machine
- Create VM/SP directory entries for OS/VS virtual machines
- Access the OS/VS system residence volume
- Ensure that the proper I/O devices are attached to the OS/VS virtual machine
- IPL and operate OS/VS under VM/SP

System Generation Recommendations

When generating OS/VS to run in a virtual machine, an installation should have these primary objectives:

- To have all commonly used transient routines made residence in storage
- To run all jobs (if possible) as V=R jobs

To meet these objectives, an installation needs to consider how it generates both VM/SP and OS/VS.

For example: OS/VS2 Release 1 (referred to as Single Virtual Storage or SVS) can use two address spaces:

- One address space for V=V jobs (that is, jobs executing even though some of their pages are not in real storage)
- Another address space for SVS and any V=R jobs

By defining the SVS virtual machine large enough for all jobs to run V=R, SVS does not have to alternate between address spaces. It should perform better under VM/SP than if it had to alternate between address spaces.

VM/SP RECOMMENDATIONS

When generating VM/SP for an OS/VS virtual machine, note the following recommendations:

VM/SP Saved Systems

IPL time can be reduced by saving any operating system after the generated operating system has been load on VM/SP. For more information about generating saved systems, refer to the VM/SP System Programmer's Guide.

Handshaking for VS1

VS1 Release 4 and subsequent releases use VM/VS handshaking. VM/VS handshaking permits instructions issued by VS1 in a virtual machine to be processed directly by the processor. It also permits VM/SP to simulate privileged instructions. For further details, refer to the topic "VM/VS Handshaking for VS1" in this section.

IPL Command Enhancement

VM/SP, via the new ATTN parameter of the CP IPL command, can pass a virtual machine console attention interrupt to an OS/VS1 virtual machine. This activates FASTNIP processing and initializes the OS/VS1 virtual machine. The ATTN parameter simulates the process of hitting the attention key (or equivalent). If the VS1 virtual machine issues the command in the CMS PROFILE EXEC, an automatic IPL of OS/VS1 utilizing FASTNIP results without operator intervention. The following process shows how to implement this new function:

1. Set up an AUTOLOG1 virtual machine in the directory. This machine logs on automatically after CP initialization. The AUTOLOG1 directory entry must contain an IPL CMS statement.
2. Issue the CP AUTOLOG command for the VS1 virtual machine in the CMS PROFILE EXEC of the AUTOLOG1 virtual machine.
3. The directory entry of the VS1 virtual machine must contain an IPL CMS statement. The CMS PROFILE EXEC for the VS1 virtual machine specifies the IPL command, with the ATTN parameter, for the VS1 SYSRES volume.
4. OS/VS1 FASTNIP activates automatically and VS1 initialization completes without operator intervention. The message IEA785I indicates FASTNIP is active and the message IEE048I reveals FASTNIP completion. The VS1 system is ready to accept jobs via the card reader.

Notes:

1. You can manually issue the IPL command with the ATTN parameter or in an EXEC, but you cannot specify it in the directory because the format is invalid.
2. The VSE environment does not need an attention interrupt to automate the IPL process.
3. If the VS1 virtual machine is autologged, console output messages are lost:
 - Unless you have started console spooling
 - Designated a secondary user to receive the console output from the disconnected VS1 virtual machine

OS/VS RECOMMENDATIONS

When generating OS/VS to run in a virtual machine, note the following recommendations:

Specifying Options

Very often, options that improve performance on a real machine have no effect (or possibly an adverse effect) in a virtual machine. For example: SEEK separation, which improves performance on the real machine, is redundant in a virtual machine. VM/SP itself issues a stand-alone SEEK for all disk I/O.

When VM/SP is the primary operating system and another operating system, such as VS1, is running one or two partitions in a virtual machine under it, generate the other operating system with as few options as possible, particularly when several virtual machines share the same system residence volume.

When VM/SP is not the primary operating system and the other operating system is usually run without VM/SP, generate the other operating system to:

- Be transparent to the users of the other system
- Have the required number of partitions or regions

Sharing the System Residence Volume

Sharing the system residence volume avoids the need to keep multiple copies of the operating system online. The shared system residence volume should be read-only. To share OS/VS among users, remove all data sets with write access from the system residence volume.

OS/VS1 RECOMMENDATIONS

When generating VS1 to run in a virtual machine, note the following recommendations:

VS1 Storage Limits

The hardware storage size used by VM/SP may be larger or smaller than the hardware storage sizes supported by VS1. In the VM/SP directory for the VS1 user, use the USER statement to define the minimum and maximum virtual storage sizes for the VS1 virtual machine. For the VS1 system, generate it to support the hardware storage size of the real machine on which the VS1 system is to run.

For example: In the USER directory statement, specify the minimum and maximum storage sizes supported for the VS1 virtual machine. Define the minimum storage size as 1 megabyte -- the minimum storage size supported by VS1. Define the maximum storage size according to the VS1 release level (such as VS1 Release 6, which has a 16 megabyte storage limit in nonpaging mode and an 8 megabyte storage limit in paging mode). For the VS1 system to run more efficiently in the VS1 virtual machine, generate its real storage size for 2 megabytes -- the storage size of the real processor. Thus, this system generation specification establishes an initial VM/SP virtual storage size of 2 megabytes even though the minimum VM/SP storage definition was 1 megabyte.

By using VS1 in nonpaging mode, VM/SP handles the paging and CCW translation for VS1, thereby eliminating the VS1 overhead associated with these functions. For a description of VM/VS handshaking between VS1 and VM/SP, refer to the topic "VM/VS Handshaking for VS1" later in this section.

VS1 in a V=R Virtual Machine

When running VS1 in a V=R machine, a user can avoid data transfer operations into real page zero by doing one of the following:

- If the VS1 nucleus already exists, replace the existing ORDER statement with the following linkage editor control statement in the VS1 linkage editor deck:

```
ORDER IEAAIH00,IEAIOS00 (P)
```

Use all INSERT control statements that were produced during the original VS1 system generation process.

- Generate a new VS1 nucleus prior to stage II execution and perform IOS alignment by modifying the ORDER control statement in the VS1 stage II job stream.

VM/VS HANDSHAKING FOR VS1

VM/VS handshaking is a communication path between the control program (CP) component of VM/SP and VS1 Release 4 (and subsequent releases) running as a virtual machine under VM/SP.

To improve their operation with VM/SP, systems generated to use VM/VS handshaking can run both in a real machine and in a virtual machine. In a virtual machine, systems that have VM/VS handshaking can more realistically simulate the operation of their real machine.

VM/VS handshaking consists of:

- Closing CP spool files when job output is complete. This function allows VM/SP to immediately process these output files without operator intervention.
- Processing pseudo page faults. When the pseudo page fault handling portion of handshaking is active, one task can be dispatched while another is waiting for a page to be brought into real storage.
- Providing a nonpaging mode to eliminate duplicate paging.
- Providing a way to avoid a PCI (programmed-controlled interruption) in a BTAM autopoll CCW loop.
- Providing miscellaneous enhancements when running under VM/SP.

Activating VM/VS Handshaking for VS1

Although handshaking is a system generation feature for VS1, it is active only when VS1 is run under the control of VM/SP; it is disabled when that same VS1 operating system is run on a real machine. The VM/VS handshaking feature is active when:

- VS1 is generated with the VM/SP option.
- The virtual machine storage space is at least 1 megabyte.

Note: If nonpaging mode is used, refer to the subtopic "VS1 Nonpaging Mode" described later in this discussion.

When loading a VS1 virtual machine that has the handshaking feature, the VS1 initialization routines determine whether the handshaking feature is available. VS1 checks for VM/SP by issuing an STIDP (store processor ID) instruction. When STIDP returns the version code X'FF', VS1 is running under VM/SP.

On receiving version code X'FF', VS1 issues a DIAGNOSE code X'00' instruction to store the VM/SP extended-identification code. If VM/SP returns a code to VS1, VM/SP supports handshaking; otherwise, VM/SP does not support handshaking.

The pseudo page fault portion of handshaking may be turned on and off with the CP SET PAGEX command. When using the CP IPL command again, PAGEX is turned off.

Pseudo Page Faults

A page fault is a program interruption that occurs when a page marked "not in storage" is referred to by an instruction within an active page. VM/SP places the virtual machine operating system referring to the page in a wait state while the page is brought into real storage. Without handshaking, VM/SP places the entire VS1 virtual machine in page wait until the needed page is available.

However, with handshaking, a multiprogramming (or multitasking) VS1 virtual machine can dispatch one task while waiting for a page request to be answered for another task. VM/SP passes a pseudo page fault (program interruption X'14') to VS1. When VS1 recognizes the pseudo page fault, it places only the task waiting for the page-in page wait and can dispatch any other VS1 task. Thus, when VS1 uses pseudo page faults, its execution under the control of VM/SP more closely resembles its execution on a real machine.

When a page fault occurs for an VS1 virtual machine, VM/SP ensures (1) that the pseudo page fault portion of handshaking is active and (2) that the VS1 virtual machine is in EC mode and enabled for I/O interruptions. Then, VM/SP reflects the page faults to VS1 by:

- Storing the virtual machine address that caused the page fault at location X'90' (the translation exception address)
- Reflecting a program interruption (interruption code X'14') to VS1
- Removing the VS1 virtual machine from page and execution wait

When VS1 recognizes program interruption code X'14', it places the associated task in a wait state. VS1 can then dispatch other tasks.

When the requested page is available in real storage, VM/SP reflects the same program interruption to VS1, except that a bit in the translation exception address field is set to one to indicate completion. VS1 removes the task from page wait, and the task is then eligible to be dispatched.

Note: VS1 with handshaking is not designed to run in a VM/SP virtual machine that is itself running under VM/SP. Otherwise, when a pseudo page fault occurs, the second-level VM/SP system (VM/SP under VM/SP) does not recognize the page fault and terminates with a PRG020 abend code.

VS1 Nonpaging Mode

When both VM/SP and VS1 support handshaking, full handshaking results when VS1 runs in nonpaging mode. However, handshaking does not require nonpaging mode. When VS1 is run under the control of VM/SP, it executes in nonpaging mode if:

- Its virtual storage space is equal to the storage space of the VM/SP virtual machine.
- Its virtual machine storage space is at least 1 megabyte.
- VM/VS handshaking is available.

The VS1 virtual machine operator initiates nonpaging mode by issuing the CP SET PAGEX ON command.

Provided that the above conditions are satisfied when VS1 is loaded under control of VM/SP, VS1 issues a special message after the IEA760A SPECIFY VIRTUAL STORAGE SIZE message:

IEA788I NON-PAGING MODE OF OS/VS1 UNDER VM/370

When VS1 executes in nonpaging mode, it uses fewer privileged instructions and avoids duplicate paging. The VS1 nucleus initialization program (NIP) fixes all VS1 pages to avoid duplicate paging.

Note: The VM/SP working set size (the estimated number of real storage pages that a virtual machine needs to execute) may be larger for a VS1 virtual machine in nonpaging mode than for one not in nonpaging mode.

Considerations unique to nonpaging mode are:

- Responding to virtual storage size message

An EOB or U response to the VS1 SPECIFY VIRTUAL STORAGE SIZE message automatically sets the VM/SP virtual storage space equal to the VS1 virtual machine storage space, provided the latter is 1 megabyte or larger.

- Nonpaging mode storage limits

Storage limits for nonpaging mode are the same as for VS1 itself:

- VS1 Release 6 has a 16 megabyte storage limit in nonpaging mode and an 8 megabyte limit in paging mode (the limit is equal to the real machine size or configuration as supported by VS1). During IPL, VS1 issues a message that specifies these limits.
- VS1 Release 5 has a 4 megabyte storage limit and issues a message to that effect.
- VS1 Release 4 has a 4 megabyte storage limit, but issues no message to that effect.

- Providing a minimum size VS1 nucleus

A minimum size VS1 nucleus tends to be more suitable for the nonpaging mode. It provides more space for problem program partitions.

- VS1 storage mapping unaffected by nonpaging mode

In nonpaging mode, VS1 maps virtual storage normally; that is, it puts these functions in the high addresses of virtual storage: the JES buffer pool, VTAM workspace, RTAM area, JES routines, resident modules, and the pageable supervisor. By minimizing the virtual storage requirements for these functions, VS1 can provide more problem program partition space.

- VS1 paging data set not used in nonpaging mode

The VS1 paging data set is not used in nonpaging mode. The VS1 system does not require page parameters and page packs.

Note: If a VS1 standalone dump should be taken, do not dump VS1 page data sets at MSGHMD021A. If an attempt is made to dump page data sets, a VM/SP abend PTR007 may occur.

- Executing V=R jobs in nonpaging mode

It is possible to execute V=R jobs in nonpaging mode. The V=R line is a valid IPL parameter and the V=R logic within VS1 is applicable. However, any benefits from such operation would appear to be questionable because the entire VS1 system functions V=R in nonpaging mode.

- FASTNIP forcing of nonpaging mode

FASTNIP automatically forces nonpaging mode when the virtual machine storage space is 1 megabyte or larger.

Note: Although handshaking does not require nonpaging mode, there may be occasions to run a VS1 system in a virtual machine with a storage space of 1 megabyte or larger and not use nonpaging mode. To avoid initiating nonpaging mode, specify the virtual storage space explicitly to VS1 in response to message IEA760A; for example: R00, '6144'.

The following examples show whether nonpaging mode is initiated when initializing VS1 in a VM/SP environment with handshaking.

Example 1:

```
VM/SP Size = 768K
OS/VS1 Virtual Storage Size = 6M
IEA760A Response = 'EOB'
```

Nonpaging mode is not initiated, and the virtual machine storage space is less than 1M.

Example 2:

```
VM/SP Size = 1M
OS/VS1 Virtual Storage Size = 6M
IEA760A Response = 'U'
```

Nonpaging mode is initiated, and VS1 forces the virtual machine storage space to 1 megabyte. The VS1 system may not complete initialization because of insufficient virtual storage space.

Example 3:

```
VM/SP Size = 4M
OS/VS1 Virtual Storage Size = 4M
IEA760A Response = R00, '4096'
```

Nonpaging mode is initiated, and the explicit response happens to equal virtual machine storage space.

Example 4:

```
VM/SP Size = 3M
IEA760A Response = R00, '6144'
```

Nonpaging mode is not initiated. The explicit response sets virtual storage space to 6 megabytes which VS1 requires for paging space.

Closing CP Spool Files

When VM/VS handshaking is active, VS1 closes the CP spool files when the job output from the VS1 DSO, terminator, and output writer is complete. Once the spool files are closed, VM/SP processes them and sends them to the real printer or punch without operator intervention.

During its job output termination processing, VS1 issues DIAGNOSE code X'08' instructions to pass the CP CLOSE command to VM/SP for each CP spool file.

VS1 Without Handshaking

When a nonhandshaking copy of VS1 (prior to Release 4.0) is run under VM/SP, the storage considerations are the same as if VS1 were running in native mode. The VS1 virtual storage space must be at least 512K larger than the virtual machine storage space.

The VS1 virtual storage space is specified at VS1 system generation time and can be altered in response to the IPL message IEA760A SPECIFY VIRTUAL STORAGE SIZE. The virtual machine storage space is specified initially in the VM/SP directory and can be altered by using the CP DEFINE STORAGE command.

BTAM Autopoll CCW Change Detection

When an operating system in a virtual machine has enabled the BTAM autopoll feature, it notifies VM/SP (via a DIAGNOSE instruction) whenever the autopoll CCWs are modified. VM/SP then modifies the real CCWs and does not check the autopoll CCWs for modifications each time the string is executed. This CCW change detection reduces VM/SP overhead and thereby improves the overall performance.

Note: If the autopoll feature is disabled for VM/SP (by the SET AUTOPOLL OFF command), a performance degradation occurs.

Miscellaneous Enhancements

When VS1 without handshaking is run in the VM/SP environment some duplication of function results. Because VS1 must perform certain functions when it is run on a real machine, it continues to perform all those functions in a VM/SP virtual machine, even though VM/SP also provides the services. However, with handshaking, VS1 avoids using many instructions and procedures that are redundant or less efficient in the VM/SP environment.

In either paging or nonpaging mode, VS1 avoids using:

- ISK (insert storage key) and SSK (set storage key) instructions; instead, VS1 uses a protection key table
- Seek separation for 2314 direct access devices
- The ENABLE/DISABLE sequence in the VS1 I/O supervisor (IOS)

- TCH (test channel) instructions preceding SIO instructions
- PCI (program-controlled interruptions) in the BTAM autopoll CCW sequence

In nonpaging mode, VS1 with handshaking avoids using:

- LRA (load real address) and RRB (reset reference bit) instructions (this is especially important when virtual machine assist is not enabled)
- The DIAGNOSE code X'10' instruction to release virtual pages or discontinuous storage
- VS1 paging and CCW translation

IBM 3850 MASS STORAGE SYSTEM CONSIDERATIONS

There are no special system generation requirements when generating OS/VS (OS/VS1 or OS/VS2 MVS) to use the MSS and operate in a virtual machine. Any VS1 or MVS system that supports the MSS can use VM/SP MSS support. VM/SP MSS support allows a VS1 or MVS virtual machine to:

- Use a dedicated mass storage control (MSC) port (or channel interface) and dedicated 3330V devices

--and--

- Act as the host for the VM/SP communicator program (which communicates 3330V mount and demount orders and responses between the MSC and VM/SP)

However, this support requires each 3330V address defined in OS/VS to be identical to the 3330V address defined both to VM/SP and to the virtual machine using it.

For details about how to generate VM/SP to support the MSS and to install the VM/SP communicator program in either VS1 or MVS, refer to VM/SP Planning and System Generation Guide. For details about how VM/SP communicates with the MSS and uses it as well as how to provide backup and recovery for MSS volumes, refer to VM/SP System Programmer's Guide. For details about MSS initialization, refer to VM/SP Operator's Guide.

Generating OS/VS Under VM/SP

This topic discusses the major steps for generating OS/VS in a virtual machine under VM/SP. While the following procedures are for generating VS1, they are similar enough to SVS and MVS procedures to be used as examples. The primary difference between generating VS1 and SVS or MVS under VM/SP is VM/VS handshaking for VS1. Handshaking requires that a user specify the VM option in the VS1 SCHEDULE macro instruction.

In the DOS/VS in a virtual machine section, the alternating technique (using one virtual machine and multiple alternating systems) was used to generate DOS/VS. While convenient for DOS/VS where only one system could be executing at any one time, it is not a convenient technique for generating OS/VS. OS/VS system generation runs are much longer, and installations should use other methods that improve overall throughput.

In this topic, two virtual machines are used to generate OS/VS under VM/SP. Figure 16 shows the directory entries for these two virtual machines: OSVSSYS and OSCMS.

The OSVSSYS virtual machine performs the majority of the system generation steps. In addition to the devices specified in the directory entry for user OSVSSYS, this user temporarily needs a dedicated tape drive at virtual address 181 on which to mount the starter system and distribution library tapes. Attaching this tape drive is explained later in this topic.

The OSCMS virtual machine is for preparing job streams for the OSVSSYS virtual machine, executing some steps in the system generation, and scanning output from OS/VS. By using the CMS editor, a user can create, store, and update the job streams used in the system generation runs. At the various stages of generation, the user can (by using the VM/SP spool system and CP spool file commands) load the appropriate job streams into the virtual card reader for OS/VS.

```
-----  
| USER OSVSSYS PASSWORD 768K  
| ACCOUNT ACCTNO BIN16  
| OPTION ECMODE REALTIMER  
| CONSOLE 01F 3210  
| SPOOL 00C 2540 R  
| SPOOL 00D 2540 P  
| SPOOL 00E 1403  
| MDISK 350 3330 0 404 DLIBA1 MR RPASS WPASS  
| MDISK 351 3330 C 404 DLIBA2 MR RPASS WPASS  
| MDISK 250 3330 0 404 OSVSSYS MR RPASS WPASS  
|  
| USER OSCMS PASSWORD 320K  
| ACCOUNT ACCTNO BIN16  
| CONSOLE 01F 3210  
| SPOOL 00C 2540 P  
| SPOOL 00D 2540 P  
| SPOOL 00E 1403  
| LINK CMSSYS 190 190 RR  
| MDISK 191 3330 50 10 UDISK1 WR RPASS  
|  
-----
```

Figure 16. Virtual Machines for OS/VS System Generation

PREPARING FOR OS/VS SYSTEM GENERATION

Before using the starter system and distribution system libraries for system generation, do the following:

- Initialize the volumes (DLIBA1 and DLIBA2) that are to contain the starter system and distribution libraries.
- Restore the starter system from tape to a DASD volume (DLIBA1).
- Use the starter system to copy the distribution libraries from the tape volumes (DLIBT1 and DLIBT2) to a DASD volume (DLIBA2).
- Create stand-alone card decks for independent utilities.
- List the VTOC of each current system volume (this listing can be used when reallocating data sets in the future).
- Initialize the volume that is to contain the new OS/VS.

One or Two Terminals?

The following examples in this topic assume that the two virtual machines (whose directory entries were shown in Figure 16) each have their own 3270 terminals. If two terminals are available, this is the most efficient method. Each system (CMS and OS/VS) runs independently with full terminal access.

However, a user can run the same procedures and use a single terminal. For example: To access the OSVSSYS virtual machine, place the OSCMS virtual machine in disconnect mode and logon to OSVSSYS. The disconnected virtual machine (OSCMS) continues to run. When ready to exchange virtual machines again, disconnect from the OSVSSYS machine and log back onto OSCMS. Logging back onto a disconnected virtual machine stops its execution. To resume operation, enter the CP BEGIN command.

Notes:

1. Before using the disconnect method of running two virtual machines from one terminal, note the disconnection considerations described later in this section.
2. Though one machine is disconnected, you can still run both virtual machines concurrently from one terminal by using the single console image facility.

INITIALIZING THE STARTER SYSTEM VOLUME

The utility control statements supplied as card input on a stand-alone system can be generated as CMS files in a VM/SP environment. The CMS editor and EXEC facilities, along with the VM/SP spool file system, allow a user to create, update, and store entire job streams or individual job steps. Data as well as job control language statements can be included in these CMS files.

For example: A user can use the OSCMS virtual machine to create the reader input required by the IBCDASDI utility program. To initialize the DASD volume to contain the starter system, logon to the OSCMS virtual machine, load CMS, and enter the following:

```
edit initstrt ojcl           (open and identify new file)
input                       (enter input mode)
initvoll job
  msg todev=3210,toaddr=01f
  dedef todev=3330,toaddr=350,valid=scratch,passes=0
  vld newvalid=dliba1,ownerid=sysprogmr
  vtocd strtadr=14,extent=5
end
(null line)                 (to return to edit mode)
file
```

The IBCDASDI control statements now reside in a CMS file. They are identified as INITSTRT OJCL and belong to the OSCMS virtual machine.

To transfer a copy of this file to the virtual reader of the OSVSSYS virtual machine, enter the following commands:

```
cp spool punch to osvssys
punch initstrt ojcl (noheader)
```

These commands queue the punched file in the virtual reader of user OSVSSYS, whether OSVSSYS is logged on or not. If OSVSSYS is logged on and does not have its messages disabled (SET MSG OFF), a message at the terminal notifies him of the additional reader input. If OSVSSYS is not logged on, the notification message is displayed as part of the user's LOGON messages when he eventually logs on.

To perform the initialization, use a second terminal and logon to userid OSVSSYS. Send the following messages to the system operator:

```
msg op pls mount scratch 3330 pack as 350
msg op pls mcunt osvs1 starter tape as 181
```

In response to these messages, the system operator mounts a 3330 volume on some available disk drive, such as 150, and enters:

```
attach 150 to osvssys as 350
```

Similarly, after mounting the VS1 starter tape on an available tape drive, such as 471, he enters:

```
attach 471 to osvssys as 181
```

The user receives the following two messages to confirm the operator's action:

```
DASD 350 ATTACHED
TAPE 181 ATTACHED
```

From the OSVSSYS terminal the user can now enter these CP commands to load the IBCDASDI program:

```
ready 181
ipl 181
```

When loaded, IBCDASDI is the first file on the starter tape. After being loaded, the program goes into a wait state. To check for this wait state condition, display the PSW by issuing:

```
#cp display psw
```

To define the control statement input device (the virtual reader) to the IBCDASDI program, press the ENTER key (or its equivalent) to display this message:

```
IBC105A DEFINE INPUT DEVICE
```

Reply, in uppercase:

```
INPUT=2540,00C
```

When the volume initialization is complete, the user receives this message at his terminal:

```
IBC163A END OF JOB
```

After displaying this message, the system enters the wait state.

RESTORING THE STARTER SYSTEM TO DISK

As performed for the IBCDASDI program, a user at the OSCMS terminal can create a CMS file for the IBCDMPRS program and punch it to the virtual reader of the OSVSSYS virtual machine. The starter tape is now positioned at the end of the first file and ready to load the second file, the IBCDMPRS program. Use the OSVSSYS terminal to enter the command:

```
ipl 181
```

To define the control statement input device to the program, follow the same procedure as in the volume initialization step. When the user receives the END OF JOB message, the starter system has been restored to the DASD volume and can now be made operational.

LOADING THE STARTER SYSTEM

To start the interim starter system, enter the following command from the OSVSSYS terminal:

```
ipl 350 clear
```

When the user receives this message:

```
IEA760A SPECIFY VIRTUAL STORAGE SIZE
```

Enter a null line if the virtual machine size is less than 384K. For the 768K machine used in this discussion, enter:

```
R 00,'2048'
```

Continue replying to the prompting messages as directed in the appropriate system generation publication.

When the user receives this message:

```
IEE048I INITIALIZATION COMPLETED
```

Ready the starter system to process input, and enter the following commands to start normal system operation:

```
MN JOBNAMES,T  
START RDR,00C  
START WTR,00E  
START INIT.PO,,,A
```

RESTORING DISTRIBUTION LIBRARIES TO DISK

Ask the system operator to mount a scratch pack as virtual address 351 to contain the distribution libraries:

```
#cp msg op pls mount scratch 3330 pack as 351
```

At the OSCMS terminal, create the IEHDASDR control statement file for initializing the volume. Then, punch the file to the OSVSSYS machine for execution. Before executing the IEHDASDR utility program, vary the DASD volume offline.

After receiving the completion message:

```
IEH806I ANALYSIS OF DDNAME=SYSIN IS COMPLETE.  
      VOLUME SERIAL NO.=DLIBA2
```

Vary the initialized volume back online.

The IEBCOPY utility program is used to unload the distribution libraries from tape to disk. The IEBCOPY control statements are in the form of load procedures and are contained in the first file of each distribution library tape.

Have the system operator mount the distribution library tapes and follow the procedures in the appropriate system generation publication to unload the distribution libraries to either one or two DASD volumes (depending upon their capacity).

CREATING CARD DECKS FOR UTILITY PROGRAMS

To punch out the independent utilities and IPL text, consider storing the card images in the OSCMS virtual machine.

Before running the IEBTPCH program, issue the CP command:

```
#cp spool punch to oscms
```

This command sends the punched output to the OSCMS virtual reader. When the IEBTPCH program has completed, enter the CP command:

```
#cp close punch
```

This command releases the file to the OSCMS virtual machine. At the OSCMS console, enter:

```
readcard utility output
```

This command creates a CMS file that contains all the card images punched. By using the CMS editor, create separate files for each utility, add appropriate control statements, and file them for future use.

INITIALIZING THE SYSTEM RESIDENCE VOLUME

Initialize the new system residence volume from the OSCMS machine while starting the stage I processing on the OSVSSYS machine. Use the CMS editor to create an IBCDASDI control statement file, and identify it as IBCDASDI CTLS. Use the EDIT subcommand GETFILE to insert the IPL TEXT file (previously punched from the distribution library volume). Then, perform the initialization by entering:

```
cp spool punch to * cont  
punch ipl ibcdasdi (noheader)  
punch ibcdasdi ctls (noheader)  
cp spool punch nocont  
cp close punch  
cp ipl 00c
```

and replying to the prompting messages.

VM/SP UTILITY PROGRAMS

A user should also consider the VM/SP utility and service programs. Many of these programs perform functions that are practically identical to the OS/VS independent utilities:

- DDR: The VM/SP DASD dump restore (DDR) service program can be obtained by running it in stand-alone mode.
- IBCDASDI: The VM/SP IBCDASDI program is available on the system disk. It has a filename of IPL and a filetype of IBCDASDI. An added feature of the VM/SP version is its ability to initialize virtual disks (minidisks).
- ICAPRTBL: The VM/SP ICAPRTBL program is available by using the CP commands LOADBUF and LOADVFCB. Use the class D command LOADBUF to load the universal character set buffer (UCSB) or forms control buffer (FCB) for specific real printers. (For details about the LOADBUF command, refer to the VM/SP Operator's Guide.) Also, use the class G command LOADVFCB to load an FCB for specific virtual printers. (For details about the LOADVFCB command, refer to VM/SP CP Command Reference for General Users.)

For a description of the DDR and IBCDASDI programs and their operating procedures, refer to the VM/SP Operator's Guide.

STAGE I PROCESSING IN THE CMS VIRTUAL MACHINE

To process stage I, the CMS virtual machine can be used to:

1. Create the stage I input job stream (job control statements and system generation macro instructions).
2. Execute a single assembly.
3. Check the assembly listing for errors.

If any errors occur, repeat all three steps of the stage I procedure.

Note: The stage I assembly can actually be executed in either the OS/VS or CMS virtual machine, whichever is more convenient.

Preparing Stage I Input

Use the CMS editor to create and store the stage I input. The job stream can then be punched to the OSVSSYS machine's virtual reader for assembly under the OS/VS starter system. The CMS editor also allows the user to edit the job stream, make changes to it, and file the modified copy under a different name.

To perform the assembly in the CMS virtual machine, follow these steps:

1. The CMS virtual machine (OSCMS) must have access to the supervisor macro instructions in SYS1.AGENLIB. These macro instructions reside on volume DLIBA2, owned by the OSVSSYS virtual machine. To have access to these macro instructions, enter the following commands from the OSCMS terminal:

```
cp link osvssys 351 351 rr
acc 351 b
filedef cmslib disk osgenlib maclib b dsn sys1 agenlib
global maclib osgenlib
```

These commands can be placed into a CMS EXEC and given control by use of a single command. For details about creating EXEC procedures, refer to the VM/SP CMS User's Guide.

2. The CMS assembler requires two special input files:
 - One CMS file to contain the job control statements that precede the system generation macro instructions. File it with a unique filename, such as STG1IN OJCL.
 - One CMS file to contain: the system generation macro instructions, an END statement, and a /* card. File it with a unique filename, such as STG1IN, and the required filetype of ASSEMBLE. (The filetype identifies it as an assembler source statement file.)

Stage I Execution

With both requirements in step 2 met, assemble the stage I input in the CMS virtual machine by issuing the CMS command:

```
assemble stg1in
```

The assembly output is these two CMS files:

```
STG1IN TEXT      (punch output)
STG1IN LISTING   (printer output)
```

The punched output is the job stream for stage II input. The printed output documents the expansion of each specified macro instruction including the punch statements that comprise the input to stage II.

When assembling under CMS, CMS displays assembly errors on the terminal. A user does not have to wait for the listing to be printed. Before retrying the assembly, use the CMS editor to correct the source file. When the assembly is finally error-free, the two output files which reflect the final assembly can then be printed, punched, or used to create job streams.

Stage I Execution in the OS/VS Virtual Machine

To punch the required job stream for stage I execution to the OS/VS machine's virtual card reader, enter the following commands:

```
cp spool punch to osvssys cont
punch stg1in ojcl (noheader)
punch stg1in assemble (noheader)
cp spool punch nocont
cp close punch
```

By using the CONT operand in the SPOOL PUNCH command, a user concatenates multiple CMS files into one punch output file.

If, at the OS/VS machine, a user enters this CP command before executing stage I:

```
#cp spool punch to oscms
```

and subsequently enters:

```
#cp close punch
```

VM/SP transfers the stage I output to the OSCMS virtual card reader. At the OSCMS terminal, issue this command:

```
readcard stg1in text
```

This CMS file corresponds to the punched output of the stage I assembly (STG1IN TEXT) performed in the OSCMS machine. VM/SP directs the assembly listing, output, the absence of any printer spooling changes, to the real in the absence of any printer spooling changes, to the real printer.

STAGE II PROCESSING

The stage I output is a job stream that contains a number of individual jobs. These jobs set up system data sets, assemble nucleus modules, and copy modules from the distribution libraries to the new system residence volume.

Preparing Stage II Input

If the three object module utility data sets used by the assembler during stage II have not been allocated and cataloged, a user can do this by creating the appropriate job stream as a CMS file. Give it a unique filename, such as STG2IN DSALLOC.

To transfer the stage II input to the OS/VS machine, issue these commands:

```
cp spool punch to osvssys cont
punch stg2in dsalloc (noheader)
punch stg1in text (noheader)
cp spool punch nocont
cp close punch
```

Stage II Execution

Process the stage II input job stream on the OSVSSYS virtual machine. To enhance this virtual machine's performance, use the various VM/SP performance options, such as running in the virtual=real area and using favored execution with a guaranteed percentage of processor time.

While the OSCMS virtual machine is not directly involved in the stage II processing, it can be used to subdivide the stage I output into individual job streams. This subdividing may make it easier to restart stage II, if necessary.

FINAL HOUSEKEEPING

The new system residence volume now contains the new system control program. Add program products in separate runs by following the procedures contained in the appropriate program product publications. Use the OSCMS virtual machine to set up or modify job streams used to test programs, even if these programs aren't part of the installation verification procedure supplied with the system control program.

Sample OS/VS Directory Entries

The following directory entries represent some batch type virtual machines that can be used to run production jobs under OS and OS/VS. The operands specified on the OPTION control statements reflect the requirements of the particular system being used. Disk space can either be dedicated or shared with other systems.

An MFT Virtual Machine

```
USER OSUSER PASSWORD 512K
  ACCOUNT ACCTNO BIN5
  IPL 230
  OPTION REALTIMER ISAM
    CONSOLE 01F 3215
    SPOOL 00C 2540 R
    SPOOL 00D 2540 P
    SPOOL 00E 1403
    DEDICATE 230 OSRES
    DEDICATE 231 OSWRK
    DEDICATE 185 285
    DEDICATE 186 286
```

A VS1 Virtual Machine

```
USER OSVUSER1 PASSWORD 512K
ACCOUNT ACCTNO BIN6
IPL 350
OPTION REALTIMER VIRT=REAL ECMODE
  CONSOLE 01F 3215
  SPOOL 00C 2540 R
  SPOOL 00D 2540 P
  SPOOL 00E 1403
  SPOOL 012 3505
  SPOOL 002 3211
  LINK VMSYS 190 190 RR
  MDISK 191 3330 21 10 UDISKA WR RPASS WPASS
  MDISK 350 333C 0 100 VOSDOS MW
  MDISK 351 3330 51 50 UDISK1 W
```

Another VS1 Virtual Machine

```
USER OSVUSER2 PASSWORD 512K
ACCOUNT ACCTNO BIN7
IPL 350
OPTION REALTIMER ECMODE ISAM
  CONSOLE 01F 3215
  SPOOL 00C 2540 R
  SPOOL 00D 2540 P
  SPOOL 002 3211
  LINK VMSYS 190 190 RR
  MDISK 191 3330 31 10 UDISKA WR RPASS WPASS
  LINK OSVUSER1 350 350 MW
  MDISK 351 3330 0 50 UDISK3 W
```

An MVS Virtual Machine (for running test jobs only)

```
USER MVSVIRT PASSWORD 4M 8M G
ACCOUNT ACCTNO BIN7
IPL CMS
OPTION ECMODE ISAM BM
  CONSOLE 01F 3215
  SPOOL 00C 2540 R
  SPOOL 00D 2540 P
  SPOOL 00E 1403
  SPOOL 00F 1403
  MDISK 191 3350 544 005 VMS005 MW PASSWORD
  LINK COMMON 190 190 RR
  LINK COMMON 191 19E RR
```

Another MVS Virtual Machine (for running test jobs only)

```
USER MVSID PASSWORD 3M 16M BCG
ACCOUNT ACCTNO BIN8
IPL CMS
OPTION REALTIMER ECMODE BM
  CONSOLE 01F 1052
  SPOOL 01C 2540 READ A
  SPOOL 01D 2540 PUNC A
  SPOOL 007 3211      A
  SPOOL 008 3211      A
  SPOOL 017 3211      A
  SPOOL 018 3211      A
  SPOOL 01E 1403      A
  SPECIAL 2FF 3270
  SPECIAL OFF TIMER
(This entry then uses the MVS
machine's PROFILE EXEC to attach
the appropriate volumes and IPL MVS.)
```

Accessing OS/VS

This topic assumes that OS/VS for use under VM/SP has already been generated and that the system residence volume is available on a real disk or minidisk in read/write status.

As the OS/VS operator, the OS/VS user needs to know the location of the system residence volume. Its location can be defined in the virtual machine configuration in one of three ways:

- Define the system residence volume as a read/write disk in the directory entry for the OSVS userid. Such a definition may appear as follows:

```
MDISK 250 3330 0 403 VS2RES WR YOUR NAME
```

Many installations prefer this approach for maintaining a directory definition of the system residence volume.

- Use the CP LINK command to define the system residence volume after logon. For example: If the SVS or MVS system programmer "owns" the system residence volume and keeps it in his virtual machine at virtual address 150, the OSVS user could gain access to it with this CP command:

```
link vs2sysp 150 250 w [name]1
```

In this command VS2SYSP is the programmer's userid, and NAME is the write password.

¹If an installation is using password-on-the-command-line suppression, a user cannot specify the password on the same command line. Passwords must be entered in such a way that they are either not displayed on display terminals or typed upon a mask for typewriter terminals.

- Have VM/SP system operator (or any class B user) exclusively attach the entire system residence volume to the OSVS userid by issuing this command:

```
attach 152 to csvs as 250
```

In this command, 152 is the real device address on which the system residence volume is mounted.

Using Virtual Devices

When using OS/VS in a virtual machine, the user is the OS/VS operator. This user must have the following devices, which are normally defined in the VM/SP directory entry:

- A virtual card reader, from which OS/VS reads the OS/VS input job stream.
- A virtual printer, which handles the printed output generated by OS/VS.
- The virtual punch, which receives punched output generated during OS/VS operation.

In addition to these unit record devices, the OS/VS operator can attach virtual tape and direct access storage devices to the virtual machine (by using either the ATTACH or DEFINE commands). The user can also specify these devices in the VM/SP directory entry.

Depending upon how OS/VS was generated, a user may need to change a virtual device address. For example: If OS/VS expects a 3211 printer at device address 002 and the directory entry does not contain this assignment, define one with the CP DEFINE command:

```
define 3211 002
```

Before using OS/VS, find out from the OS/VS system programmer what are the installation's virtual device requirements.

DEFINING THE OPERATOR'S CONSOLE

The operator's console must be at the address specified during OS/VS system generation. The easiest way to ensure this is to define the appropriate console address in the directory entry. For example: The CONSOLE directory control statement could appear as follows:

```
CONSOLE 01F 3210
```

This statement defines a virtual 3210 console at virtual address 01F.

USING THE VM/SP SPOOL FILE SYSTEM

A user should let the VM/SP spool file system handle printer or punch output that does not have to be printed or punched. For example: When using the alternating technique, route print output to the virtual card reader by using this CP SPOOL command:

```
#cp spool printer to *
```

After issuing this command, a user can subsequently load the CMS system and create a CMS file from the data in the virtual reader (by using the CMS READCARD command). Then, by using the CMS editor, the user can scan the contents of this data at his terminal.

Preparing Jobs for an OS/VS Virtual Machine

Prepare and submit a job stream to an OS/VS virtual machine in one of two ways:

- Place a deck of real punched cards that contain the appropriate job control, program and data in the real card reader. Place a CP ID statement at the beginning of this job stream deck to indicate the OS/VS userid; for example:

```
ID OSVS
```

```
-- or --
```

```
USERID OSVS
```

Either statement is a valid ID statement for directing the input that follows to the OSVS user's virtual reader (the reader with the lowest virtual device address).

- Use the CMS system to create a CMS file containing images of what would normally be submitted through a card reader on a real System/370. Enter the CP SPOOL command to cause subsequent punched output to be directed to the virtual card reader of the OS/VS machine. Enter the CMS PUNCH command to generate the virtual card deck:

```
cp spool punch to osvs  
punch vsjob27 jcl (noheader
```

The NOHEADER option of the PUNCH command suppresses punching a CMS READ control card at the beginning of the deck.

A job stream spooled to OS/VS by either of these methods remains in the card reader of the OS/VS virtual machine until the user starts an OS/VS reader.

When spooling jobs to a virtual machine, clear any data that may remain in the virtual punch from previous jobs by issuing these CP commands:

```
cp spool punch nocont  
cp close punch purge
```

These commands ensure that the virtual punch is purged of any existing reader files. The first command is required if the punch had been originally spooled with the CONT operand.

JOB ENTRY AND OUTPUT RETRIEVAL

When running OS/VS in a virtual machine, a primary consideration is job entry and output retrieval. Several techniques can be used for these functions:

- Use the OS/VS virtual machine in batch mode where it operates as OS/VS in native mode. It reads in job streams through a dedicated card reader and prints output generated by the virtual machine on a dedicated printer.
- A single directory entry (userid) can contain a configuration sufficient for running both CMS and OS/VS (the alternating technique). Load CMS to create and edit OS/VS job streams and to check the OS/VS output. Load OS/VS to run the OS/VS job streams.
- Use two different userids to keep two virtual machines running. (This is the optimum environment.) Use one userid for the OS/VS machine while using the other for a CMS machine to create job streams and inspect output. By using the CP DISCONN command, users can run both virtual machines from the same terminal. Thus, both machines are running concurrently, but a user communicates with only one at a time. If two terminals are available, each system can run independently of the other.

Loading OS/VS

At logon, use the userid for running OS/VS. It may be a user's own userid when running alternating or concurrent systems. It may be a special userid that an installation has set aside exclusively for the OS/VS machine.

```
logon osvs
```

Because extended control (EC) mode is required to operate OS/VS, the directory entry should contain the ECMODE specification on the OPTION control statement. If it is not included, enter this CP command to enable extended control mode simulation:

```
set ecmode on
```

Note: To run OS/PCP and OS/MFT, a user does not need the ECMODE option. To run OS/MVT, a user does not need the ECMODE option unless running the generalized trace facility (GTF) under OS/MVT. However, OS/PCP, OS/MFT, and OS/MVT normally require the REALTIMER option.

At this point, between logging on and loading the operating system, a user may find it desirable or necessary to alter the virtual machine's storage size. For example: If the directory entry specifies 1 megabyte of storage and a user needs 2 megabytes for this particular terminal session, issue the CP DEFINE command:

```
define storage as 2m
```

Virtual machine storage can be redefined up to the limit set in the USER control statement of the directory entry.

Once the IPL volume is made available, enter this command to load OS/VS:

```
ipl 250
```

OS/VS responds with a "SPECIFY SYSTEM PARAMETERS" message. The proper use of a system parameter list (IEASYS00 or IEASYSxx), created during or subsequent to the OS/VS system generation process, can result in a significant saving of time. Commonly used operator commands can be placed in the SYS1.PARMLIB data set to shorten the IPL process.

OS/VS Operation

To control OS/VS, use OS/VS operator commands to hold and release queues and jobs and to start initiators or define partitions. Users can observe the progress of the command's execution by following the OS/VS messages. Figure 17 shows how VM/SP loads VS1 in a virtual machine.

```

| LOGON AT 01:96:14 EST WEDNESDAY 10/29/75
| def stor 3072k
| STORAGE = 03072K
| q set
| MSG CN , WNG ON , EMSG TEXT , ACNT ON , RUN OFF
| LINEDIT ON , TIMER REAL , ISAM OFF , ECMODE ON
| ASSIST ON SVC , PAGEX OFF , AUTOPOLL OFF
| IMSG ON , AFFINITY OFF
| def C09 01f
| CONS 01F DEFINED
| q chan
| CHANNELS = SEL
| def chan bmx
| CHANNELS = BMX
|
|   The virtual selector channels have been redefined
|   to block multiplexer. This is a performance
|   consideration to improve the processing of I/O
|   operations. To avoid issuing the CP DEFINE BMX
|   command, the VM/SP directory entry can specify
|   the BMX option in the OPTION control statement.
|
| i 250
| IEA760A SPECIFY VIRTUAL STORAGE SIZE
| r 00,'3072'
|
|   Specifying the size of the virtual machine's
|   storage relative to the VS1 virtual storage
|   size is a performance consideration. VS1 has
|   been shown (in performance studies) to operate
|   more efficiently in a virtual environment if it
|   is not forced to do any of its own paging.
|   Also, in nonpaging mode VS1 avoids many
|   privileged instructions, thereby reducing
|   VM/SP overhead.
|
|   To force VS1 into a nonpaging mode, use VM/VS
|   handshaking and define the storage size for the
|   virtual machine equal to the virtual storage
|   size requested by VS1.
|
| IEA788I NON-PAGING MODE OF VS UNDER VM/SP
| IEE054I DATE=75.300,CLOCK=009.00.00
| IEE054I DATE=75.300,CLOCK=009.00.02,GMT
| IEA764I NIPNULL,CMDMON,DFNPARM,JESNULL,,,,,SETPARM,,
| IEA101A SPECIFY SYSTEM AND/OR SET PARAMETERS FOR RELEASE 04.0 OS/VS1
| r 00,'u'
| IEA106I IEAAPF00 NOT FOUND IN SYS1.PARMLIB

```

Figure 17. Sample IPL of VS1 under VM/SP (Part 1 of 2)

```

| IEE140I SYSTEM CONSOLES
|   CONSOLE/ALT CMD AUTH      ID      ROUTCP
|   01F/01F      M  ALL      01      1-10,10-16
| IEF031I SYSGEN VALUES TAKEN FOR JES
| IEF866I DEFINE COMMAND BEING PROCESSED
| IEE804I P0=(C=AOB,576'K,A,I),P1=(C=AOB,576K,A,E,LAST),
| IEE804I P2=(INACTIVE),P3=(INACTIVE),
| IEE804I P4=(INACTIVE),P5=(INACTIVE),
| IEE804I P6=(INACTIVE),P7=(INACTIVE),
| IEE543I 250K BYTES FREE SPACE
| IEE817I TMSL=NONE
| IEE805I DEFINITION COMPLETED
| IEE101A READY
| IEE009I JLPRM=(U)
| IEE050I MN JOB NAMES,T
| IEE048I INITIALIZATION COMPLETED -

```

Figure 17. Sample IPL of VS1 under VM/SP (Part 2 of 2)

When using CMS, initial operator commands can be incorporated into a CMS EXEC procedure as part of the job stream. For example: To create the following EXEC procedure called SETUPVS, issue these commands:

```

CP LINK VSSYS 250 250 RR OSPASS
CP DEFINE 009 AS 01F
CP IPL 250

```

After starting the appropriate OS/VS readers, the virtual machine is ready to receive input from card readers, DASD, or tape drives.

COMMUNICATING WITH CP

During OS/VS virtual machine operation, a user can issue CP commands to (1) communicate with the VM/SP system operator or other virtual machine users, and (2) query and alter the status of the configuration and spool files. In general, a user can enter any of the CP commands normally permitted under his userid's privilege class.

Entering CP commands while an OS/VS virtual machine is running depends on the terminal mode (as defined by the CP TERMINAL command or its default value). When not running as the VM/SP system operator, the default terminal mode is VM. In this mode, pressing the attention key (or its equivalent) once passes an interruption pending condition to the OS/VS virtual machine. Pressing the attention key twice places the virtual machine in the CP command environment from which CP commands can be entered. For a complete description about how to use the attention key, refer to VM/SP CP Command Reference for General Users.

Using the #CP Function

In most cases during virtual machine operation, a user can use the #CP function to enter CP commands directly from the virtual machine. If the virtual machine has issued a read to the terminal, enter a CP command with the #CP function. For example: When no longer using virtual tape drive 397 mapped to real tape drive 492, issue:

```

#cp detach 397
#cp msg op i am done with real drive 492

```

VM/SP immediately processes these command lines, and the virtual machine read remains outstanding.

Note: A user may not always be able to enter CP commands with the #CP function. The read issued by OS/VS at the terminal must be for at least as many bytes as entered in the #CP command line; any additional information is truncated. If the read is for at least three bytes, enter the #CP command. This command places the user in CP command mode from which CP commands can be entered directly. To return to the virtual machine environment, enter the BEGIN command.

Using OS/VS in Batch Mode Under VM/SP

When many users submit jobs to a single OS/VS virtual machine, someone is generally needed to tend the machine as an operator. The virtual machine operator must make those decisions required of an operator on a real machine; that is, deciding what work is going to be done and what is the most efficient way of doing it.

In batch mode, one user runs as the OS/VS machine (userid OSVS) and other users (like CMSID1) may submit jobs either through the virtual card reader, through the system card reader, or through JES remote stations. If the card reader is not dedicated to the OS/VS virtual machine, place an ID card at the beginning of each job stream, such as:

```
USERID OSVS A
```

In this card: the USERID (or ID) indicates the valid beginning of an ID card, OSVS is the name of the VM/SP user to receive the card input in his virtual reader, and A is the VM/SP reader class.

Note: If the VM/SP system operator has dedicated a card reader to the OS/VS virtual machine, then the ID card must be omitted at the beginning of the deck.

A user can send jobs to the OS/VS machine from other virtual machines by spooling his punch to the OS/VS userid, such as:

```
#cp spool punch to osvs
```

Entering this statement causes subsequent punched output to appear in the virtual card reader for userid OSVS.

Alternating Between CMS and OS/VS Under VM/SP

When working in a program development environment (rather than a production environment) and unable to test programs directly under CMS, a user can alternate between OS/VS and CMS in a single virtual machine. Some advantages of this technique (described in this topic) are:

- Reduced unit record output. Users can examine program output and compiler listings online, check the results, and resubmit the job without producing any output on the system unit record devices.
- Faster turnaround time (generally) than in a batch environment.

Before using this technique, users should be familiar with the CMS editor and CMS file manipulation commands found in the VM/SP CMS User's Guide.

LOADING CMS INTO A VIRTUAL MACHINE

To load CMS into a virtual machine, use the CP IPL command and specify either a saved system name or a device address:

```
ipl cms
-- cr --
ipl 190
```

When CMS responds with a message like this:

```
CMS VERSION 3.0
```

Enter the CMS commands to create an OS/VS job stream.

USING THE CMS EDITOR TO PREPARE JOB STREAMS

The following CMS procedure creates an OS/VS job stream that can be passed to the OS/VS virtual machine's reader. It shows how to compile a PL/I program under OS/VS, making the PL/I source file available as a CMS file called PLI27 DECK.

edit pli127 jcl	open a CMS file by name
input	go into input mode
//pli127 job cps,fred,msglevel=1	enter jcl entries
//cat exec plifc	
//sysin dd *	▼
(null line)	return to edit mode
getfile pli127 deck	copy over PL/I source file
input	go into input mode
/*	enter jcl entries
//	▼
(null line)	return to edit mode
file	write the file to disk

ISSUING SPOOL COMMANDS TO CONTROL UNIT RECORD DEVICES

Spool the virtual punch to the virtual machine:

```
cp spool punch to *
```

This command causes subsequent punched output to appear in the user's own virtual card reader. To submit a job to the OS/VS machine, punch the JCL and associated card data.

Spool the virtual printer to the virtual card reader:

```
cp spool printer to *
```

Instead of routing printed output to the real printer, this command causes it to appear in the virtual card reader. By using the CMS machine, each print file can then be read, examined, and either purged or printed at the real printer.

Note: Since a user may find both punch and printer files in the virtual reader, consider using the spool file class attribute to control which files are to be processed at any one time.

PUNCHING CMS FILES

Use the CMS PUNCH command to transfer the job stream to the virtual card reader of the OS/VS virtual machine:

```
punch pli27 job (noheader
```

By specifying NOHEADER option of the PUNCH command, VM/SP does not punch a CMS READ control card at the beginning of the output deck.

INITIALIZING OS/VS

Load OS/VS into the virtual machine by issuing this command:

```
cp ipl 250
```

When an OS/VS reader is started, it reads the job stream that had been previously punched with the punch spooled to the user's own userid. For example, the OS/VS command:

```
s rdr,00c
```

starts a reader on virtual device 00C and reads those cards that appear in the OS/VS reader queue.

RELOADING CMS INTO A VIRTUAL MACHINE

When the job stream has been processed, reload CMS and use the READCARD command to create a CMS file from the printed output and put it onto a CMS disk. The output can now be examined with the CMS editor or TYPE command. When hardcopy output is needed, the file can be printed via the CMS PRINT command. When using the READCARD command to create a CMS file, do not use a filetype of LISTING. Otherwise, VM/SP assumes the first character of each line to be a control character, which is not printed.

EXAMINING OS/VS VIRTUAL MACHINE OUTPUT

To examine a job's output executed under the OS/VS virtual machine, spool the unit record output to the user's own userid. Now read the file in the virtual card reader onto a CMS disk by using CMS READCARD command:

```
readcard pli27 job
```

The CMS TYPE command or editor can then be used to scan such a file.

If programming errors occurred during the execution of the job stream, the CMS editor can be used to make corrections to the source program, correct job control statements, or resolve any other execution problems. After making these corrections, resubmit the job stream.

Dynamic SCP Transition to or from Native Mode

Prior to VM/SP support for dynamic SCP transition to or from native mode, it was inconvenient for installations to transfer control of an operating system from VM/SP virtual machine mode to native mode or vice versa. They had to shutdown their operating system and IPL it again. With the transition function, installations can make such a transition without shutting down the operating system and initializing it again. Thus, installations avoid the overhead associated with continuously running an operating system under VM/SP but can still have VM/SP function when it is needed.

VM/SP support for making a dynamic SCP transition to or from native mode is primarily for SVS and MVS operating systems. However, the VS1 operating system run without VM/VS handshaking can also use this support.

VM/SP makes the transition from VM/SP to native mode (or vice versa) as transparent to the operating system user as possible. Before making this transition, the operating system must run in a V=R virtual machine with dedicated I/O devices. This transition function requires the V=R mode of operation because the storage occupied by the operating system under VM/SP and the storage used in native mode must have the same addresses. The transition function requires dedicated devices because the I/O devices must have the same addresses while running under VM/SP (virtual I/O addresses) as when running without VM/SP (real I/O addresses).

Restriction for MVS Virtual Machines

The transition function does not support either real or virtual channel reconfiguration. Thus, for an MVS system to operate in a virtual machine and make a dynamic transition to native mode, do not generate the system with channel reconfiguration hardware (CRH) support; that is, do not specify OPTIONS=(CRH) in the MVS CTRLPROG system generation macro.

You can generate MVS to run in a virtual machine with TCAM active. However, you cannot request the transition to native mode if TCAM was generated with the VM/370 option. Specifying the VM/370 option with TCAM forces MVS to issue DIAGNOSE instructions when running in a virtual machine environment. This can cause disastrous results when MVS tries to execute the DIAGNOSE instructions in a native environment after transition. Therefore, TCAM cannot be generated with the VM/370 option if the virtual MVS system desires dynamic transition.

OPERATING PROCEDURES

To transfer control of an operating system from under VM/SP to native mode, follow these steps:

1. All VM/SP users must logoff, except for the VM/SP system operator and the V=R virtual machine.
2. The V=R virtual machine user must:
 - a. Ensure that the virtual device addresses of the dedicated devices have the same real I/O addresses.
 - b. Ensure that there are no minidisks.
 - c. Stop all virtual spooling devices in the virtual machine.
 - d. Issue the CP CLOSE command to close any open spool files.
 - e. Issue the CP DETACH command to detach these virtual spooling devices.
 - f. Ensure that the interval timer is turned off (by using the CP SET TIMER OFF command) if it is not used by the virtual machine.
 - g. Ensure that no activities are being traced and no address stops are being set.
3. The VM/SP system operator must drain the real unit record devices.

Note: The transition to native mode cannot take place as long as any outstanding I/O events exist for dedicated devices that belong to the MVS virtual machine; this includes teleprocessing lines.

When these conditions are met and VM/SP is running in uniprocessor mode, the VM/SP system operator can issue the CP class A command QVM userid to give the V=R virtual machine control of the processor. However, if an installation doesn't want to return from native mode to VM/SP, they can have the VM/SP system operator issue the QVM userid command with the NORETURN operand.

Note: When the operating system is in native mode, the operating system user:

- Has normal use of the restart PSW key. It can be used to present a restart interruption to the native operating system.
- Must not alter the storage above the VM/SP V=R storage area. This storage contains the VM/SP nucleus, which VM/SP requires to transfer control of an operating system from native mode back to VM/SP.

Before transferring control of an operating system from native mode back to VM/SP virtual machine mode, the operating system user must ensure that the I/O devices and their addresses are identical to those previously used under VM/SP. Once these conditions are met, the VM/SP system operator must follow these steps at the system console to transfer operating system control from native mode to VM/SP virtual machine mode:

1. Display the restart PSW at main storage location zero.
2. Use steps 2a and 2b to determine whether the instruction address in the PSW is for VM/SP or the native operating system. (The instruction address is in the last three bytes of the PSW.)
 - a. For VM/SP, if the address points to either entry point DMKQVMRS or DMKQVMRX in module DMKQVM, the operator can proceed to step 3. (The entry point addresses are in the CP load map produced either by the system generation process or by the installation whenever it changes the CP nucleus.)
 - b. For the native operating system, the address points to the system's restart interrupt handler when the system has done a PSW refresh. The VM/SP system operator must change this address to either entry point DMKQVMRS or DMKQVMRX (as listed in the CP load map described for step 2a). Enter the address for DMKQVMRS when the operating system does not use the OS/VS2 MVS/System Extensions Program Product, 5740-XE1. Otherwise, enter the address for DMKQVMRX.
3. Subtract eight bytes from the instruction address in this PSW.
4. Locate the storage area pointed to by the address determined in step 3 and store X'FF' in the first byte of that location. Do not change the other three bytes.
5. Press the restart key to return control to VM/SP.

Note: If VM/SP was generated to support an AP system, VM/SP can resume AP operations in the attached processor when the system resource operator (class B) varies it online.

ERROR RECORDING

When an SCP makes the transition to native mode, error records for the SCP are in two locations:

- When under VM/SP, VM/SP records them in its error recording cylinders.

--and--

- When in native mode, the SCP records them in its SYS1.LOGREC data set.

To find all the error records that pertain to the SCP, the user must look in both locations. To put the records in chronological sequence, the user can follow the time and date recorded in each record.

Using More Than One Virtual Machine

A user can run multiple systems, each in its own virtual machine and each controlled from its own terminal. However, if multiple terminals are not available, and the single console image facility is not used, a user can use one terminal to control all systems but only one at any one time.

This approach combines the alternating system and batch techniques. Separate userids are used to run OS/VS in one virtual machine and to prepare jobs and examine OS/VS output in a CMS virtual machine.

After submitting a job stream under the CMS userid, issue the DISCONN or LOGOFF command (depending upon how soon the user intends to logon to the CMS ID again). The terminal is now free to be used for running the OS/VS job stream under the OSVS userid:

```
logcn cmsid
.
.
.
(route jobs to OSVS user)
.
.
.
disconn
logon osvs
.
.
.
(run jobs)
```

The procedures for running with two virtual machines are much the same as those used in running a single virtual machine in alternating mode. The primary difference is that a user now spools the virtual punch and printer to the other virtual machine instead of spooling them to the user's own userid:

```
logcn cmsid
sp pun osvs
.
. (route jobs to OSVS user)
.
disconn
logcn osvs
#cp sp prt cmsid
#cp sp pun cmsid
.
. (run OS/VS jobs)
.
```

DISCONNECTION CONSIDERATIONS

When using more than one userid to alternate communications between operating systems, consider:

- How OS/VS may read additional jobs from the card reader?
- What happens when a read is issued at the disconnected OS/VS virtual console?
- What happens to the console output of the disconnected virtual machine?
- Are you using the single console image facility to control both virtual machines concurrently from one terminal?

Sending Jobs to a Disconnected OS/VS Machine

When using CMS to route jobs to a disconnected OS/VS virtual machine, spool the OS/VS reader with the CONT operand of the CP SPCOL command:

```
spool reader cont
```

This command allows the OS/VS reader to read more than a single job at a time without operator intervention.

Console READs and WRITEs in a Disconnected OS/VS Machine

- Without the single console image facility:

When running OS/VS disconnected, a 15-minute time-out begins when a console read occurs. If the virtual machine does not respond to the read before the 15 minutes elapse, VM/SP automatically logs off the virtual machine.

Whenever running OS/VS disconnected, it is suggested that a console log be created. This log provides a user with a history of what jobs were run. It can also indicate any unusual circumstances that occurred during the terminal session.

To start console spooling, issue:

```
spool cons start
```

To stop console spooling and to print the log, issue:

```
spool console stop close
```

When a virtual machine is running disconnected, all console output is lost unless a user initiates console spooling, such as by issuing:

```
#cp spool console start
```

Spooling of the console output continues until a user either logs off or issues:

```
#cp spool console stop
```

Disconnecting the virtual machine does not stop console spooling. Therefore, the spooled console log for a terminal session, punctuated with several disconnects, consists of one uninterrupted printer file.

- With the single console image facility:

If a disconnected virtual machine with an active secondary user issues a read to the console, a message is sent to the console informing the secondary user. No 15-minute time-out is initiated. The secondary user then satisfies the read by issuing a SEND command to the disconnected virtual machine.

Any console output from a disconnected virtual machine with an active secondary user will appear on the console of the secondary user. Each output line will have a prefix consisting of the disconnected virtual machine's userid followed by a colon. Also, console spooling can be used as described in "Without the single console image facility".

Developing and Testing Programs to Run in an OS/VS Virtual Machine

The previous discussions demonstrated how the CMS editor and EXEC facility can help a user prepare jobs for execution in an OS/VS virtual machine. In addition to these CMS functions, there are a number of other CMS commands for developing and testing programs.

For example: A user can use the CMS READCARD and MOVEFILE commands to create CMS files from source programs or existing JCL that are on cards or magnetic tape. One advantage of storing source programs on CMS disks is that they can be maintained as backup copies of a program while a second version is being tested and debugged. By using the CMS editor or commands like COPYFILE, SORT, and RENAME, a user can modify and copy CMS disk files.

Refer to VM/SP CMS User's Guide for information on how to compile and execute many types of OS/VS programs under CMS.

OS/VS2 MVS Uniprocessor Under VM/SP

When operating MVS in uniprocessor mode under VM/SP, VM/SP simulates three privileged and two nonprivileged System/370 instructions.

The three privileged instructions are:

- CLRIO (clear I/O)
- IPK (insert PSW key)
- SPKA (set PSW key from address)

The two nonprivileged instructions are:

- CS (compare and swap)
- CDS (compare double and swap)

VM/SP allows the compare instructions (CS and CDS) to execute normally; that is, it does not simulate them when the real machine is equipped with the appropriate hardware feature. However, when MVS is run under VM/SP on a machine that does not have these instructions installed, VM/SP simulates them.

Use of Single Processor Mode in AP and MP Systems

In tightly-coupled multiprocessing (MP) and attached processor (AP) systems, single processor mode allows an installation to dedicate a processor to an MVS V=R virtual machine. In single processor mode, VM/SP runs in uniprocessor mode in the main processor, and the MVS V=R virtual machine runs under VM/SP in the main processor and has the exclusive use of the other processor for MP or AP operations. An MVS V=V virtual machine cannot use single processor mode. However, other virtual machines can operate under VM/SP concurrently with the MVS V=R virtual machine in single processor mode. Prior to single processor mode, MVS virtual machines could only run MVS in uniprocessor mode -- not in MP or AP mode.

OPERATING PROCEDURES

To use single processor mode, an installation must meet these four conditions:

1. The multiprocessing feature is installed on the hardware for the MP or AP system.
2. VM/SP has a V=R storage area.
3. VM/SP is running in uniprocessor mode.

Note: To run in uniprocessor mode if VM/SP was generated to support an AP system, the system resource operator (class B) must vary offline the attached processor. Once it is offline, VM/SP begins running in uniprocessor mode.

4. The system operator enables single processor mode for the VM/SP system by issuing the class A CP command:

```
spmode on
```

After VM/SP is in single processor mode, the MVS virtual machine user must IPL (or re-IPL) MVS so that it can gain control of the dedicated processor for MP or AP operations.

Note: When a user initializes the MVS V=R virtual machine before VM/SP is in single processor mode, he is initializing MVS in uniprocessor mode -- not in MP or AP mode.

In single processor mode, VM/SP simulates MP privileged instructions issued by the MVS virtual machine. It also reflects MP external interruptions to this virtual machine. When other interruptions (such as I/O) or privileged instructions occur in the main processor, VM/SP simulates them and reflects the activity to the proper virtual machine. When they occur in the dedicated processor, the MVS virtual machine handles them.

To leave single processor mode, the MVS V=R virtual machine user should first finish all MVS processing and reset the virtual machine. (The user can reset the virtual machine in several ways: by initializing CMS, by issuing the CP SYSTEM RESET command, or by logging off the virtual machine.) Once processing is finished and the virtual machine is reset, the VM/SP system operator issues the class A CP command:

```
spmode off
```

This command returns control to VM/SP in the normal uniprocessor mode of operation. If VM/SP was generated to support an AP system, VM/SP can resume AP operations in the attached processor when the system resource operator (class B) varies it online.

To determine whether the system is in single processor mode, either the system operator (class A) or a virtual machine user (class G) can issue the CP command:

```
query spmode
```

Restrictions

- When the MVS virtual machine runs in MP mode, the MVS virtual machine operator should not vary offline the main processor. Varying MVS offline in the main processor would cause these problems:
 - VM/SP cannot control the main processor.
 - VM/SP abnormally terminates when the MVS user in the dedicated processor attempts to vary the main processor online.
 - MVS in native mode (through a dynamic SCP transition to native mode as previously described in this section of this publication) cannot return and operate as a virtual machine under VM/SP.
- Single processor mode does not support either real or virtual channel reconfiguration. Thus, if an MVS system is to operate in a virtual machine and use single processor mode, do not generate the system with channel reconfiguration hardware (CRH) support; that is, do not specify OPTIONS=(CRH) in the MVS CTRLPROG system generation macro.

ERROR RECORDING

When in single processor mode, VM/SP cannot intercept SVC 76 (the error recording SVC) in the dedicated processor. Thus, error records for the MVS V=R virtual machine are in two locations: (1) the V/M error recording cylinders when SVC 76 is issued in the main processor, and (2) the MVS SYS1.LOGREC data set when SVC 76 is issued in the dedicated processor.

To find all the error records that pertain to the MVS V=R virtual machine, the user must look in both locations. To put the records in chronological sequence, the user can follow the time and date recorded in each record.

Note: Duplicate error records appear for channel checks reflected on the main processor. When MVS in the main processor issues SVC 76 for a channel check, VM/SP intercepts the SVC 76 and records the error in its error recording cylinders. However, VM/SP then reflects (or passes) the SVC 76 back to MVS for recording in its SYS1.LOGREC data set.

TAKING A VM/SP DUMP

single processor mode, when the PSW restart key is pressed, In single processor mode, when the PSW restart key is pressed, VM/SP reflects the restart interruption back to the MVS V=R virtual machine and does not take a VM/SP dump.

To take a VM/SP dump while VM/SP is in single processor mode, the system operator must follow these steps:

1. Display the restart PSW at main storage location zero.
2. Subtract eight bytes from the instruction address in this PSW. (The instruction address is in the last three bytes of the PSW.)

3. Locate the storage area pointed to by the address determined in step 2 and store X'FF' in the first byte of that location. Do not change the other three bytes.
4. Press the restart key to take the dump.

After the dump is taken, the dump program automatically reinitializes VM/SP.

Note: When CP is not in a loop or wait state, the VM/SP system operator can use the CP commands DCP and STCP (class C) to perform these steps. Otherwise, the operator must perform these steps at the console. For details about how to use the console to display and alter main storage, refer to the appropriate System/370 operating procedures publication.

Summary

When loading OS/VS into a virtual machine, the terminal becomes the OS/VS operator console, and the virtual machine user becomes the operator responsible for entering all commands and responses. The three basic techniques for running OS/VS in a virtual machine are: batch mode, alternating between OS/VS and CMS in a single virtual machine, and (for OS/VS2 users only) running OS/VS2 disconnected.

Before using one of these techniques, an installation must understand how to:

- Generate OS/VS to run in a virtual machine
- Create VM/SP directory entries for OS/VS virtual machines
- Access the OS/VS system residence volume
- Ensure that the proper I/O devices are attached to the OS/VS virtual machine
- IPL and operate OS/VS under VM/SP

The primary objectives when generating OS/VS to run in a virtual machine should be to have all commonly used transient routines resident in storage and to run all jobs V=R if possible. To meet these objectives, an installation needs to consider how it generates both VM/SP and OS/VS. (OS/VS can also be generated under VM/SP.)

To control OS/VS in a virtual machine, use OS/VS operator commands to hold and release queues and jobs, and to start initiators or define partitions. Users can observe the progress of the command's execution by following the OS/VS messages. Also, additional operator commands and control statements must be entered at the console before running jobs on the OS/VS virtual machine.

OS/VS virtual machine users can use the CMS editor and the EXEC facility to prepare jobs for execution in an OS/VS virtual machine. They can also use CMS commands to develop and test programs on CMS disks.

Index

*CP function (see CP function)

A

abend dump, printing for virtual machines 6
accessing
 devices, by VM/SP virtual machine 63
 error records, for operating systems under VM/SP 4
accessing terminals logged on via AUTOLOG facility, with secondary user's console 56-57
accounting
 ACCT option, generating job accounting cards for virtual user 46
 elapsed processor time, in DOS/VS virtual machine 81-82
ACCT option, OPTION control statement, accounting considerations unique to operating systems 46
allocating space
 CP disks, for VM/SP virtual machine 62
 for temporary disks, with CP format/allocate program 17
alternate consoles, specifying for virtual machine 14
alternate path support
 defining hardware features to VM/SP, using PDEVIC macro 18
 for virtual machines 17
 mutually exclusive with reserve/release support 18
 restriction, not for reserved devices 19
 summary of VM/SP reserve/release effect on 20
alternating between operating systems (see alternating technique)
alternating technique
 between virtual machine operating systems
 for CMS and OS/VS 141-144
 for DOS/VS and CMS 102-108
 for operating systems 23-25
application programs
 designing to run efficiently under VM/SP 5
 developing and testing
 using alternating technique between CMS and operating system 23
 using OS/VS virtual machine 149
 timer-driven, specifying REALTIMER option for 47
 VIRT=REAL option, using for 48
ASP virtual machine
 connecting with real ASP machine 31
 using virtual CTCA, to connect two virtual machines 30
attached processors, dedicating to MVS V=R virtual machine 149-152

AUTOLOG command

 defining AUTOLOG facility, in AUTOLOG1 virtual machine 54-55
 use with AUTOLOG facility 54
AUTOLOG facility 54-57
 AUTOLOG1 virtual machine
 defining for multiple systems 56-57
 defining for single system 55
 defining in VM/SP directory 54-55
 generating secondary users 54-57
 logging on users automatically, using CP AUTOLOG command 54
AUTOLOG1 (see AUTOLOG facility)
autopoll, channel program, bypassing when using BTAM under VM/SP 8

B

batch mode
 for DOS/VS virtual machine, using 101
 for OS/VS virtual machine, using 141-144
batch virtual machine
 operating systems supported under VM/SP 3
 using under VM/SP 3
BMX option, using block multiplexer channels, for virtual machine operating systems 7
BTAM, autopoll channel program, VM/SP checking and bypassing for 8

C

channel programs, ISAM, option for executing under VM/SP 7
channels
 dedicated, specifying for virtual machine 16
 model dependencies, effect on channel error recovery procedures 12-13
 overlapping SIO requests on, using BMX option 7
channel-to-channel adapter (see CTCA)
CMS (conversational monitor system)
 alternating technique
 for DOS/VS 102-108
 with OS/VS in same virtual machine 141-144
 loading for use with DOS/VS 102-103
 VM/SP component
 brief description of 2
 description of 3
CMS editor, advantages when generating operating systems under VM/SP 43
CMS/DOS, linkage editor, program execution under CMS/DOS 81
CMS/DOS environment, using instead of a guest DOS/VS or VSE system 112-113

common area, calculating for OS/VS2 virtual machines 36-38
 communicating between virtual machines
 using IUCV 8
 using VMCF 8
 communication lines, defining, by using SPECIAL control statement 53
 communications system, testing, using multiple-access virtual machines 28
 configuration (see virtual machine)
 CONSOLE control statement
 defining the operator's console, for a virtual VM/SP system 60
 examples
 defining VM/SP virtual machine 59
 specifying a secondary user 55
 specifying secondary console 13
 used to define a secondary user 51
 consoles (see multiple consoles)
 control program (see CP)
 controlling a disconnected user, via CP SEND command, from a secondary user's console 13
 controlling a disconnected user, on secondary user's console 51
 CP (control program)
 VM/SP component, brief description of 2
 CP commands
 entering
 from DOS/VS virtual machine 99
 from OS/VS virtual machine 140
 CP disks, formatting and allocating space, for VM/SP virtual machine 62
 CP function, for OS/VS virtual machine, how to use 140
 CPU (see processor)
 CTCA (channel-to-channel adapter)
 defining, by using SPECIAL control statement 53
 real, connecting real and virtual ASP systems 31
 virtual, specifying for ASP virtual machines 30

D

DASD
 for virtual machine
 dedicating to 17
 defining for 17
 sharing data between users 17
 VM/SP reserve/release support for 18
 DASD dump restore program (see DDR)
 data transfer, between virtual machines, using VMCF 8
 data transfer between virtual machines, using IUCV 8
 DDR (DASD dump restore)
 VM/SP utility program, description of 130
 debugging, printing dumps for operating systems under VM/SP 6

DEDICATE control statement
 defining remote devices, for guest systems 51-52
 examples
 assigning CTCA to ASP virtual machine 31
 defining devices with special I/O protocol (caution) 53-54
 defining virtual addresses on real TCU 29
 specifying a device as primary console 14
 unique considerations, for virtual machine operating systems 51

dedicated
 channels, specifying for virtual machine 16
 devices
 alternative to dedicated channels 16
 summary of VM/SP reserve/release effect on 21
 to use unsupported devices 4
 using DEDICATE control statement 51
 volumes, VM/SP release/release support for (specification example) 20
 dedicated processor, for MVS V=R virtual machine 149-152
 defining
 to OS/VS virtual machine
 devices 136
 operator's console 136
 virtual devices, using VM/SP directory control statements 53-54
 virtual lines, to multiple-access virtual machine 25-29
 defining a virtual console, for a second level VM/SP system 60
 DIAGNOSE instruction codes
 invalidating entries below high-water mark 34-35
 notifying VM/SP of BTAM autopoll channel programs 8
 using VMCF 8
 diagnose interface, for communication between operating systems and VM/SP 10
 dial-up terminals (see terminals)
 DIRECT program
 creating directory entries, for VM/SP virtual machine 62
 using for VM/SP virtual machine, example 66-67
 directory entry (see virtual machine)
 disconnected operation
 for DOS/VS virtual machine 111-112
 for OS/VS virtual machine 147-148
 with a secondary user's console 111-112
 with secondary user specified 147-148
 disk space, temporary, obtaining for virtual machines 17
 display-operator-console mode (see DOC)
 distribution library
 restoring
 for DOS/VS generation under VM/SP 84
 for OS/VS system generation under VM/SP 128

DMKPIO module, defining, for VM/SP virtual machine 61
 DMKSYS module, defining, for VM/SP virtual machine 61
 DOC (display-operator-console)
 mode, specifying local 3270 as primary console 15
 DOS (see also DOS/VS)
 files, using T-disk space for 17
 ISAM, using under VM/SP 47
 virtual machine, directory entry example 89-90
 DOS/VS (see also virtual machine; VM/SP)
 accounting, elapsed processor time under VM/SP 81-82
 Advanced Functions Program Product incorporating VM/SP linkage enhancements 9,78
 application programs, designing to run efficiently under VM/SP 5
 BTAM autopoll channel programs, bypassing and checking for under VM/SP 8
 disconnected operation, with a secondary user specified 111-112
 double spooling under VM/SP, eliminating by specifying DEDICATE control statement 51-52
 entering statements, with logical line end character 77
 handshaking, use of autopoll required 8
 ISAM, using under VM/SP 47
 linkage editor, program execution under CMS/DOS 81
 printing abend dumps under VM/SP 6
 processing of non-standard label cylinders, in a virtual machine 91
 running disconnected, under the control of a secondary user 77
 supervisor
 assembling and loading under VM/SP 85-88
 generating to run under VM/SP 79
 relocating page zero to run under VM/SP 48
 system generation under VM/SP
 adding IOCS modules 88
 assembling and loading a supervisor 85-88
 building (creating) CMS job stream files for 83
 copying distribution system to disk 84
 DOS/VS recommendations 79
 handshaking support 78
 recommendations 78-82
 sample virtual machine for generating 82
 using an interim system 85
 VM/SP recommendations 78
 using a secondary user's console, with CMS and the quest system 109-110
 virtual machine 77-114
 accessing CMS from more than one virtual machine 109-110
 accessing system residence volume 90
 CMS/DOS environment for (description) 112-113
 communicating with CP 99
 defining operator's console 92
 description (introductory) 77
 developing and testing programs for use under 112-113
 directory entry example 89-90
 disconnecting 109-110
 how to run more than one 108
 interrupting execution of 100-101
 IPL from a card reader 96-98
 IPL from a console 94-95
 operating under VM/SP 98
 preparing job streams for 93-94
 running in batch mode 101
 sharing system residence volume with other users 90-91
 starting a job stream 99
 technique for alternating between CMS and DOS/VS 102-108
 using unique standard label cylinders to access system residence volume 91
 using virtual unit record devices 92
 DOS/VSE (see also DOS/VS)
 system generation under VM/SP, handshaking support 78
 using as an alternate method, for sharing the system residence volume 90-91
 dump, printing virtual machine abends 6
 dump processing
 for V=R users using shadow table bypass function, page zero table entry changed 33
 for VM/SP virtual machine 63
 example 75-76
 system operator procedures when using dedicated processor 151-152
 dynamic SCP transitions
 operating procedures 144-146
 restrictions 144-145
 system operator procedures 145-146
 dynamic transition to or from native mode, description 144

 E
 ECMODE option
 directory entry example, specifying for VM/SP virtual machine 56-57
 specifying for System/370 operating systems 46
 using for operating systems run under VM/SP 7

ECPS: VM/370
effect on
 DOS/VS accounting 81-82
 virtual machine paging performance 5
 virtual machine performance 38-39
performance option, specifying for
 virtual machines 40-42

ECPS: VM/370 (extended control-program
support)

error recording
 by VM/SP, for virtual machine operating
 systems 48
 for operating systems under VM/SP 4
 when SCP makes dynamic transition to
 native mode 145-146

Error recording, when using dedicated
processor for MVS V=R virtual machine 151

error records, for operating systems under
VM/SP, accessing and recording 4

error recovery, virtual machine, provided
by operating system in 15-16

error recovery procedures, for
model-dependent processors and channels,
effect on SCPs run under VM/SP 12

EXEC procedure, for alternating between CMS
and DOS/VS 107-108

extended control mode, ECMODE option,
specifying for System/370 operating
systems 46

extended control-program support (see
ECPS: VM/370)

F

favored execution, performance option,
specifying for virtual machines 40-42

FDP (Field Developed Program), used under
VM/SP (reference) 4

Field Developed Program (FDP), used under
VM/SP (reference) 4

file names, selecting, for DOS/VS minidisks
73

first level storage, definition 6

flip-flop technique (see alternating
technique)

format/allocate program (CP)

 allocating temporary disk space, for
 virtual machines 17

 for VM/SP virtual machine, how to format
 and allocate space on CP disks 62

formatting, CP disks, for VM/SP virtual
machine 61

formatting error recording cylinders, with
SRF devices, for hardcopy reports 4

four-channel switch, VM/SP alternate path
support for 17

full screen console support, for virtual
operating systems 21-22

G

generating operating systems, under VM/SP,
advantages of 43

generation procedures, under VM/SP, for
other operating systems 43

quest real storage, same as second level
storage 6

quest virtual storage, same as third level
storage 6

H

handshaking (see VM/VS handshaking)

handshaking for VS1, VM/SP, for OS/VS 116

high-water mark

 defining for shadow tables 34-35

 using with quest systems, to improve
 performance 35

I

IBCDASDI program, initializing T-disk
space, for files 17

INDICATE command, to measure VM/SP
performance 42

initializing disks and minidisks, for
DOS/VS virtual machines 78

Installed User Program (IUP), used under
VM/SP (reference) 4

interactive problem control system (see
IPCS)

interactive virtual machine, operating
systems supported under VM/SP 3

inter-user communications vehicle (see
IUCV)

interval timer, using REALTIMER option, to
allow updating under VM/SP 7

I/O devices

 AUTOLOG facility, to automatically
 initiate for virtual machine operating
 systems 54-57

 using those unsupported by VM/SP 4

I/O interruptions, virtual machine,
processed by operating system in 15-16

IOCS modules, adding to DOS/VS virtual
machine 88

IPCS (interactive problem control system)
dump processing, for VM/SP virtual
machine 63

 VM/370 component (Release 6), brief
 description of 2

IPL

 DOS/VS virtual machine

 from a card reader 96-98

 from a console 94-95

 for VM/SP virtual machine 63-64

 example 71-75

 loading systems automatically, using IPL
 control statement 50

IPL command enhancement, VM/SP, for OS/VS
116

IPL control statement

 unique considerations, for virtual
 machine operating systems 50

 using AUTOLOG facility, to specify
 AUTOLOG command 54

ISAM

 option, using for operating systems run
 under VM/SP 7

 using under VM/SP 47

IUCV (inter-user communication vehicle)
data transfer and communication between
virtual machines, brief description and
reference 8
IUP (Installed User Program), used under
VM/SP (reference) 4

J
job accounting (see accounting)

L
LINK control statement
accessing minidisk with other users,
subject to virtual reserve/release
processing 19
accessing system residence volume, for
OS/VS virtual machine 135
example, defining VM/SP virtual machine
59
linkage editor, program execution, under
DOS/VS and CMS/DOS 81
linking
to access CMS system residence volume,
by VM/SP virtual machine 61
to CMS disks, by VM/SP virtual machine
(example) 66
loading systems automatically, by
specifying IPL control statement 50
locked pages, performance option,
specifying for virtual machines 40-42
logical data sharing, definition of 17

M
magnetic tape drives
dedicating to virtual machines, by using
DEDICATE control statement 51
defining, by using SPECIAL control
statement 53
Mass Storage System (see 3850)
MDISK control statement
defining system residence volume, for
OS/VS virtual machine 135
examples, defining VM/SP virtual machine
60
specifying virtual reserve/release
support 19
measuring, performance factors, to evaluate
virtual machine performance 38-42
MFT (see also OS/VS; virtual machine;
VM/SP)
ISAM, using under VM/SP 47
MFT virtual machine, directory entry
example 133

minidisk
accessing, by VM/SP virtual machine 64
definition of 17
file-name, selecting for DOS/VS virtual
machine 81
for virtual machine, defining 17
initializing, for DOS/VS virtual machine
78
reserve/release support, not used for
paging and spooling 19
restriction, cannot be on dedicated
channels 16
shared, VM/SP reserve/release support
for 19
sharing data between users 17
summary of VM/SP reserve/release effect
on 21
temporary, defining and using for
virtual machine 17
virtual reserve/release support,
protecting with (caution) 19
virtual VM/SP reserve/release support,
protecting with 19
mode switching, in virtual machine
environment, for quest consoles 21-22
MONITOR command, to measure VM/SP
performance 42
monitoring system activity
collecting shadow table statistics
for SET STMULTI CSEG command 36-38
for SET STMULTI n command 36-38
for SET STMULTI USEG command 36-38
to measure VM/SP performance 42
MSS (Mass Storage System) (see 3850)
multiple consoles, specifying for virtual
machine 14
multiple programming operating systems
(see also DOS/VS; OS/VS)
special considerations 9-32
multiple shadow table support, description
of 32
multiple-access virtual machine
defining and using 25-29
defining virtual addresses on real TCU
29
directory entry, how to specify virtual
TCUs 53
directory entry example 27
for testing a communications system 28
operating systems supported under VM/SP
3
performance considerations 30
using local terminals 26-27
using remote terminals 26-27
using under VM/SP 3
multiprocessing
dynamically transferring control to or
from native mode 144-146
for MVS V=R virtual machine 149-152
MVS
uniprocessor mode, VM/SP restrictions
149
virtual machine, directory entry
examples 134-135

MVS/System Extensions support, enabling by use of 370E option 50
MVT (see also OS/VS; virtual machine; VM/SP)
ISAM, using under VM/SP 47

N

nondisruptive transition to native mode, description 144
nonpaging mode, initiating, for VS1 120
non-switched lines, defining for multiple-access virtual machine 28

O

operating systems
alternating between, technique for 23 supported under VM/SP 3
operation
of DOS/VS virtual machine 98
of VM/SP virtual machine 63-64
operator console
defining
for DOS/VS virtual machine 92
for OS/VS virtual machine 136
OPTION control statement
example, defining VM/SP virtual machine 59
unique considerations, for virtual machine operating systems 46

options

specifying for
DOS/VS virtual machine 80
OS/VS virtual machine 117-118

OS (see also OS/VS)

ASP, using under VM/SP 31
files, using T-disk space for 17
ISAM, using under VM/SP 47
OS/MFT (see MFT; OS/VS; virtual machine; VM/SP)
OS/MVT (see MVT; OS/VS; virtual machine; VM/SP)
OS/PCP (see PCP; OS/VS; virtual machine; VM/SP)

OS/VS

application programs, designing to run efficiently under VM/SP 5
AUTOLOG facility, to automatically initiate OS/VS under VM/SP 54-57
double spooling under VM/SP, eliminating by specifying DEDICATE control statement 51-52
dynamically transferring control to or from native mode 144-146
ISAM, using under VM/SP 47
nucleus, relocating page zero to run under VM/SP 48
printingabend dumps under VM/SP 6
specifying a secondary user when disconnected 115

system generation under VM/SP 124-133
IPL command enhancement 116
OS/VS recommendations 117-118
preparation 125
selecting number of terminals 126
stage I execution in CMS virtual machine 131
stage I execution in OS/VS virtual machine 132
stage I input preparation 130-131
stage I processing in CMS virtual machine 130
stage II execution 133
stage II input preparation 132
stage II processing 132
starter system loading 119
starter system restoring 128
starter system volume initialization 126-127
using VM/SP utilities and service programs 129-130
virtual machines required 124-125
VM/SP recommendations 115
VS1 handshaking (major discussion) 119

system residence volume, accessing 135
transferring control to or from native mode, restrictions 144-146
using a secondary user's console 126
using a secondary user during disconnected operation 147-148
virtual machine 115-152

accessing system residence volume 135

alternating technique between CMS and OS/VS 141-144

communicating with CP 140
defining operator's console 136
defining virtual devices 136
description (introductory) 115
disconnected operation 144-146
IPL 138
IPL example 139-140
job entry and output retrieval 138
operating techniques 139-140
preparing and submitting job stream 137

restrictions when running MVS in uniprocessor mode 149
running multiple systems 144-146
using in batch mode 141-144
using spool files 137
using to develop and test programs 149

OS/VS1 (see VS1; OS/VS)

OS/VS2 (see OS/VS)

P

page fault, pseudo, affected by VS1 handshaking processing 120
page tables, shadow table bypass restrictions, for V=R users 33

page waits, VM/SP handling of, effect on
 operating systems run under VM/SP 10
 page zero
 shadow table bypass restrictions, for
 V=R users 33
 using virtual=real option, to locate for
 virtual machine operating systems 49
 paging (see also nonpaging mode)
 activity, reducing for virtual machines
 5
 factors, affecting virtual machine
 performance 5
 virtual=real option, eliminating double
 paging 48
 partition sizes, specifying for DOS/VS
 virtual machine 79
 PCP (see also OS/VS; virtual machine;
 VM/SP)
 ISAM, using under VM/SP 47
 performance
 for BTAM autopoll channel programs
 executed under VM/SP 8
 guidelines, for operating systems run
 under VM/SP 38-42
 interactive, emphasizing response time
 for 42
 I/O waits, establishing processing
 priority 10
 measurements, for both VM/SP and virtual
 machines 42
 multiple-access virtual machine,
 considerations for 30
 options
 controlled by general user 40-42
 controlled by system operator for
 system analyst 40-42
 reducing page waits for operating
 systems under VM/SP 10
 using for both VM/SP and virtual
 machines 40-42
 virtual machine, paging factors
 affecting 5
 physical pack sharing, definition of 17
 PTD tape, restoring, for DOS/VS system
 generation under VM/SP 81
 POWER, generating for DOS/VS and DOS/VSE
 virtual machines 80
 POWER/VS (see POWER)
 PP (Program Product), used under VM/SP
 (reference) 4
 preparation, OS/VS, system generation under
 VM/SP 125
 printing dumps, for virtual storage
 operating systems under VM/SP 6
 priority
 establishing for operating systems under
 VM/SP
 for I/O waits 10
 for page waits 10
 priority option, performance option,
 specifying for virtual machines 40-42
 privileged instructions
 reducing
 effect on virtual machine performance
 32
 for DOS/VS virtual machine 79
 processor
 dedicating to MVS V=R virtual machine
 149-152
 model dependencies, effect on channel
 error recovery procedures 12-13
 reserve/release support, limitation
 issuing reserve CCW commands to shared
 minidisks 19
 single, reserve/release support for 19
 VM/SP reserve/release support for,
 reserving devices between 19
 processor time, for DOS/VS virtual machine,
 accounting for 81-82
 PROFILE EXEC
 AUTOLOG facility, specifying for
 (example) 56-57
 AUTOLOG1 virtual machine, defining for
 AUTOLOG facility 54-55
 Program Product (PP), used under VM/SP
 (reference) 4
 pseudo timers, defining, by using SPECIAL
 control statement 53

 R
 RCTLUNIT macro, VM/SP reserve/release
 support, reserving devices between
 processors 19
 RDEVICE macro
 defining hardware features for VM/SP
 alternate path support 18
 defining real CTCA to ASP virtual
 machine 31
 VM/SP reserve/release support, reserving
 devices between processors 19
 real available processor storage, same as
 first level storage 6
 REALTIMER option
 considerations for operating systems run
 under VM/SP 7
 directory entry example, for VM/SP
 virtual machine 56-57
 virtual timers, permitting operating
 systems to update under VM/SP 47
 reducing
 paging activity, for virtual machines 5
 privileged instructions
 effect on virtual machine performance
 32
 for DOS/VS virtual machine 79
 releasing temporary minidisks, description
 17
 remote devices, dedicating to virtual
 operating systems, via DEDICATE control
 statement 51-52
 remote spooling communications subsystem
 (see RSCS)
 reserved devices
 specifying VM/SP reserve/release support
 between processors 19
 VM/SP reserve/release support, defining
 for 19
 reserved pages, performance option,
 specifying for virtual machines 40-42

reserve/release support
 by VM/SP for DASD units used by
 operating systems 18
 definition of 18
 for dedicated volumes, example 20
 for single processors 20
 mutually exclusive with VM/SP alternate
 path support 18
 processing for DASD with alternate path
 support 18
 RCTLUNIT macro, specifying for reserved
 devices 19
 RDEVICE macro, specifying for reserved
 devices 19
 real, summary of VM/SP reserve/release
 effect on 20
 reserved devices, VM/SP generation and
 usage considerations 19
 shared minidisks, limitation issuing CCW
 commands to 19
 summary 20
 using with VM/SP alternate path support
 18
 virtual
 definition of 18
 for users accessing same minidisk via
 LINK control statements 19
 specifying in MDISK control statement
 19
 summary of VM/SP reserve/release
 effect on 20
 response time, emphasizing performance for
 under VM/SP 42
 restoring contents of saved virtual
 machine, using IPL command 49
 restrictions
 OS/VS, when transferring control to
 native environment 144-146
 using full screen console support, in
 quest environment 21-22
 RSCS (remote spooling communications
 subsystem)
 VM/370 component (Release 6), brief
 description of 2

S

saved systems
 VM/SP
 for DOS/VS 70
 for OS/VS 116
 saving contents of virtual machine, using
 VMSAVE option 49
 second level directory (see SLD)
 second level storage, definition 6

secondary user
 accessing CMS and a virtual DOS/VS
 system without disconnecting and
 reconnecting 109-110
 alternate means for running disconnected
 OS/VS virtual machine 126
 controlling a disconnected DOS/VS
 virtual machine 77
 defined in the CONSOLE control statement
 51
 disconnection consideration 111-112
 providing services for a disconnected
 user 13
 using with an OS/VS virtual operating
 system 115
 with disconnected OS/VS virtual machine
 147-148
 secondary user defined, in conjunction with
 AUTOLOG facility 55
 segment tables
 defining for a virtual operating system,
 using SET STMULTI command 36-38
 shadow table bypass restrictions, for
 V=R users 33
 selective invalidation, of shadow tables
 below high-water mark 32
 service programs, VM/SP, using 129-130
 Service Record File (SRF) device,
 formatting error recording area, for
 printed output 4
 SET command
 enabling and disabling MVS/System
 Extensions support 50
 shadow table support
 defining concurrent tables for
 virtual machines 32,34-35
 defining high-water mark for
 selective invalidation 33
 defining high-water mark for shadow
 tables 33
 eliminating tables for V=R virtual
 machine users 33
 selecting values for STBYPASS nnnnnK
 operand 34-35
 selecting values for STMULTI operand
 36-38
 STBYPASS nnnnnK operand for V=R users
 (usage note) 33
 STFIRST option permits use of
 STBYPASS nnnnnK operand 47
 STMULTI operand for V=R users (usage
 note) 33
 specifying options, for virtual machine
 operating systems 46

shadow tables
 bypass function
 for V=V users 34-35
 specifying with SET STBYPASS command 33
 V=R user usage and restrictions 33
 V=V user usage and restrictions 34-35
 defining concurrent tables for virtual machine
 using SET STMULTI command 32,36-38
 selectively invalidating below
 high-water mark 33
 shared DASD, VM/SP reserve/release support for 18
 single console image facility (see secondary user)
 specifying a secondary user 13
 single processor mode
 enabling and disabling for VM/SP 150
 for dedicated processor
 effect on shadow table bypass function for V=R users 33
 error recording considerations 151
 VM/SP dumping procedures 151-152
 single-user virtual machines (see batch virtual machine)
 SIO instructions, reducing number executed by virtual machines 6
 SID (second level directory)
 reducing entries read, for DOS/VS virtual machine 81
 SPECIAL control statement
 examples
 defining devices with special I/O protocol (caution) 53-54
 specifying a device as primary console 15
 specifying a remote terminal as secondary console 15
 specifying virtual CTCA for ASP virtual machines 31
 unique considerations, for virtual machine operating systems 53
 specifying a secondary user
 to control a guest DOS/VS system 77
 via CONSOLE control statement 51
 specifying secondary users, for users that are automatically logged on, via AUTOLOG facility 55
 SPOOL control statement
 defining a virtual 3800 printer 51
 examples
 defining devices with special I/O protocol (caution) 53-54
 defining VM/SP virtual machine 59
 spool devices, accessing, by VM/SP virtual machine 64
 spool files
 closing, using VM/VS handshaking 123
 defining, for OS/VS virtual machine 137
 spooling
 double spooling operations, handling for VM/SP virtual machine 64
 operating systems run under VM/SP 11-12
 3800 restriction 11
 spooling devices, dedicating to virtual machines, using DEDICATE control statement 51-52
 stage I
 for OS/VS system generation under VM/SP execution in CMS virtual machine 131
 execution in OS/VS virtual machine 132
 input preparation 130-131
 processing in CMS virtual machine 130
 stage II
 for OS/VS system generation under VM/SP execution 133
 input preparation 132
 processing 132
 standard label cylinders, DOS/VS virtual machine, to access system residence volume 91
 starter system
 for OS/VS system generation under VM/SP loading/restoring 128
 volume initialization 126-127
 STFIRST option
 OPTION control statement
 permits user to issue SET STBYPASS nnnnnK command 47
 shadow table bypass note for V=R users 36-38
 storage levels in a VM/SP system, printing dumps for virtual storage operating systems 6
 storage size, specifying, for VS1 virtual machine 118
 string switch, VM/SP alternate path support for 17
 SVC interruptions, handling under VM/SP 48
 SVS (see OS/VS)
 switched lines, defining for multiple access-virtual machine 28
 SYSRES (see system residence volume)
 system generation (see DOS/VS; OS/VS; VM/SP)
 system generation recommendations, for VM/SP and OS/VS 116
 system programs, developing and testing, using OS/VS virtual machine 149
 system residence volume
 for DOS/VS virtual machine
 accessing 90
 sharing between machines 80,90-91
 for OS/VS virtual machine
 accessing 135
 sharing between machines 118

T
 tape devices, summary of VM/SP reserve/release effect on 21
 T-disks, defining and using for virtual machines 17
 teleprocessing lines, accessing, by VM/SP virtual machine 64

temporary disk space, obtaining for virtual machines 17

terminals

- dial-up, supporting with multiple-access virtual machine 28
- local, using with multiple-access virtual machine 26-27
- remote
 - used by multiple-access virtual machine 26-27
 - using virtual TCU to control (example) 29
- supporting with both VM/SP and multiple-access systems, how to connect to 28

test system (TESTSYS) virtual machine, directory entry example 56-57

testing

- communications system, using multiple-access virtual machines 28
- VM/SP in a virtual machine 60

third level storage, definition 6

time-of-day clock, VM/SP virtual machine restriction 63,71

time-sharing systems, performance, emphasizing response time 42

transferring output, between CMS and operating system, technique for 23

TSO multiple-access virtual machine, directory entry example 56-57

two-channel switch, VM/SP alternate path support for 17

two-channel switch additional feature, VM/SP alternate path support for 17

U

unit record devices

- for spooling, dedicating to virtual machine operating systems 51-52
- virtual, use by DOS/VS virtual machine 92

unsupported devices

- dedicating to virtual machines, by using DEDICATE control statement 51-52
- defining, by using SPECIAL control statement 53
- using under VM/SP 4

USER control statement, example, defining VM/SP virtual machine 59

using CP SEND command, to control disconnected users, from secondary user's console 51

using STBYPASS command, with guest systems with only one address space, to increase performance 34-35

using virtual machine assist, with MVS/SE, on 158-3 processors 50

utility programs, VM/SP, using 129-130

V

V=R (see virtual=real)

VIRT=REAL option, double paging, eliminating for virtual machine operating systems 48

virtual=real

- jobs executing under nonpaging mode, in VS1 virtual machine 120-122
- performance option, specifying for virtual machines 40-42

virtual control units, caution, specifying devices with special I/O protocol 53-54

virtual devices

- defining with VM/SP directory control statements 53-54
- states or conditions under VM/SP 15-16

virtual disk (see minidisk)

virtual interval timer assist, DOS/VS, accounting under VM/SP 81-82

virtual lines, defining for multiple-access virtual machine 25-29

virtual machine (see also VM/SP)

- adding I/O devices without using real devices, using SPECIAL control statement 53
- allocating resources 2
- alternate path support for 17
- ASP, using under VM/SP 31
- configurations
 - for alternating between operating systems 24
 - for VM/SP virtual machine 60-62
 - measuring workload performance for 33
 - performance factors for operating systems run under VM/SP 38-39
- dedicated channels, specifying 16
- dedicated devices, using as alternative to dedicated channels 16
- defining a virtual operator's console for a guest VM/SP 151-152
- defining a virtual 3800 printer, for use by guest system 51
- definition of 1
- directory entries
 - creating and updating 43-44
 - specifying control statements 46
- directory entry examples
 - AUTOLOG1 54-57
 - AUTOLOG1 for multiple systems 56-57
 - AUTOLOG1 for single system 54-55
 - CMS for use with OS/VS 124-125
 - DOS 89-90
 - DOS/VS 82,89-90
 - for alternating between operating systems 24
 - MFT 133
 - multiple-access 27
 - MVS 134-135
 - OS/VS 124-125
 - test system (TESTSYS) 56-57
 - TSO multiple-access 56-57
 - VM/SP virtual machine 59
 - VS1 134-135

double paging, eliminating by use of
 VIRT=REAL option 48
 error recovery, provided by operating
 system in 15-16
 full screen support for guest console
 21-22
 I/O interruptions, processed by
 operating system in 15-16
 IUCV, communicate/transfer data between
 8
 multiple-access, defining and using
 under VM/SP 25-29
 operating systems supported under VM/SP
 3
 options
 defining 40-42
 optional services for operating
 systems run under VM/SP 7
 specifying 7
 performance
 guidelines for operating systems run
 in 38-42
 measuring for 42
 printingabend dumps for operating
 systems 5
 reducing VM/SP paging activity for 5
 SIO instructions, reducing I/O
 operations for 6
 specifying consoles for 14
 testing and updating VM/SP virtual
 machine 60
 using as a secondary user's console 77
 using IPL command to restore contents
 49
 virtual devices
 defining with VM/SP directory control
 statements 53-54
 VM/SP I/O management for 15
 VMCF, to communicate/transfer data
 between virtual machines 8
 virtual machine assist
 effect on virtual machine paging
 performance 5
 performance option, specifying for
 virtual machines 40-42
 STFIRST option, permits issuing of SET
 STBYPASS nnnnnK command 47
 using with MVS/System Extensions 50
 virtual machine, effect on performance
 38-39
 virtual machine communication facility
 (see VMCF)
 Virtual Machine/System Product (see VM/SP)
 virtual TCU, example, controlling remote
 terminals 29
 virtual timer, updating under VM/SP 47
 VMA (see virtual machine assist)
 VMAP (VM/370 Performance/Monitor Analysis
 Program)
 measuring workload performance of both
 VM/SP and virtual machines 40
 MONITOR command, reduce and analyze data
 from 42
 shadow tables
 reducing and printing monitor records
 for 34-35, 36-38
 VMCF (virtual machine communication
 facility)
 data transfer and communication between
 virtual machines, brief description and
 reference 8
 VMFLOAD, using for VM/SP virtual machine,
 example 68
 VMSAVE option, OPTION control statement,
 automatically saving contents of virtual
 machine 49
 VM/SP (Virtual Machine/System Product)
 (see also virtual machine)
 alternate path support, for virtual
 machines 17
 alternating between CMS and operating
 system, to create and test programs 23
 ASP, using in virtual machine 31
 BTAM autopoll channel programs, checking
 and bypassing for 8
 channel checks, handling under VM/SP
 12-13
 components, brief description 2
 enhanced 3270 support, control 3270 by
 DIAGNOSE code 10
 error recording, for operating systems
 4
 error records, accessing for operating
 systems 4
 full screen support, display size by
 DIAGNOSE code 10
 generating operating systems under,
 advantages of 43
 I/O instructions, reducing number issued
 by virtual machine 6
 I/O management, sharing devices between
 virtual machines 15
 I/O waits, effect of processing on
 operating systems under VM/SP 10
 multiple-access system, using in virtual
 machine 25-29
 page waits, effect of processing on
 operating systems under VM/SP 10
 paging
 effect on application programs 4
 reducing for virtual machines 5
 performance
 measuring for 42
 paging factors affecting virtual
 machines 5
 performance additions (description)
 40-42
 performance factors, for operating
 system run in virtual machine 40-42
 performance guidelines 38-42
 performance options, for system analyst
 and general user 40-42
 providing disconnected services via
 secondary user's console 13
 providing full screen console support,
 for guest operating systems 21-22
 reserve/release support, for operating
 system in virtual machine 18
 service programs, using 129-130
 shadow table maintenance support 32-38
 activate TOD clock interface by
 DIAGNOSE code 10
 MVS low-address protection by
 DIAGNOSE code 10

- spooling, closing spool files 12
- system generation
 - defining alternate path support for 18
 - defining CTCA for ASP virtual machine 31
 - defining hardware features for alternate path support 18
- unsupported devices, how to use 4
- utility programs, using 129-130
- virtual devices, processing I/O operations for virtual machine operating systems 15
- virtual storage preservation support via DIAGNOSE code 10
 - VMSAVE option 49
- VM/VS handshaking, operating systems supported by 9
- VM/SP linkage enhancements for DOS/VS
 - via Advanced Functions Program Product 9,78
- VM/SP options, defining, for virtual machine operating systems 46
- VM/SP virtual machine
 - accessing CMS 61
 - accessing devices 63
 - configuration for 60-62
 - verifying (example) 65-76,70,71-75
 - CP disks, formatting and allocating 62
 - creating directory, example 66-67
 - directory entry example 59
 - dump processing, using IPCS 63
 - IPL, time-of-day clock restriction 63
 - IPL under VM/SP 63
 - operation under VM/SP 63-64
 - spooling, handling double spooling operations 64
 - system generation, defining DMKSYS and DMKRIO modules 61
 - terminal session example 65-76
 - testing and updating 60
 - 2305 devices, using for paging 61
- VM/VS handshaking
 - autopoll, required for DOS/VS 8
 - for DOS/VS and DOS/VSE 70
 - ISAM option required for 7
 - VM/SP performance, effect on 40-42
- VS1
 - enhancements in nonpaging mode 123
 - major discussion 119
 - using nonpaging mode 120-122
- VM/370 Performance/Monitor Analysis Program (see VMAP)
- VSAM, files, using T-disk space for 17
- VSE environment (see DOS/VSE)

- VS1 (see also OS/VS; virtual machine; VM/SP)
 - BTAM autopoll channel programs, bypassing and checking for under VM/SP 8
 - handshaking, using without 123
 - nonpaging mode
 - initiation examples 120-122
 - VM/SP considerations 120-122
 - nucleus, nonpaging mode considerations 120-122
 - system generation under VM/SP
 - recommendations 119
 - VM/SP recommendations 119
 - 3850 MSS considerations 124
 - virtual machine, directory entry examples 134-135
 - VM/VS handshaking, functions provided for virtual machine 119
- VS2 (see OS/VS; virtual machine; VM/SP) Release 1 (see OS/VS)

- W
- workload
 - performance factors, for operating systems run under VM/SP 33
 - virtual machine, definition of 40
- 2
- 2305
 - dedicating to virtual machines, using DEDICATE control statement 51-52
 - for VM/SP virtual machine, paging usage (discussion) 61
- 3
- 3380, formatting restriction with IBCDASDI program 17
- 370E, using with virtual machine assist 50
- 370E option, OPTION control statement, enabling/disabling MVS/System Extensions support 50
- 3800
 - defining as a virtual printer, for guest operating systems 51
 - restriction when used as virtual printer, by guest operating system 11
- 3850, VS1, system generation considerations for 124

This manual is part of a library that serves as a reference source for systems analysts, programmers, and operators of IBM systems. This form may be used to communicate your views about this publication. They will be sent to the author's department for whatever review and action, if any, is deemed appropriate. Comments may be written in your own language; use of English is not required.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation whatever. You may, of course, continue to use the information you supply.

Note: *Copies of IBM publications are not stocked at the location to which this form is addressed. Please direct any requests for copies of publications, or for assistance in using your IBM system, to your IBM representative or to the IBM branch office serving your locality.*

- | | Yes | No | | |
|---|--------------------------|----------------------------|--------------------------|--|
| • Does the publication meet your needs? | <input type="checkbox"/> | <input type="checkbox"/> | | |
| • Did you find the material: | | | | |
| Easy to read and understand? | <input type="checkbox"/> | <input type="checkbox"/> | | |
| Organized for convenient use? | <input type="checkbox"/> | <input type="checkbox"/> | | |
| Complete? | <input type="checkbox"/> | <input type="checkbox"/> | | |
| Well illustrated? | <input type="checkbox"/> | <input type="checkbox"/> | | |
| Written for your technical level? | <input type="checkbox"/> | <input type="checkbox"/> | | |
| • What is your occupation? | | | _____ | |
| • How do you use this publication: | | | | |
| As an introduction to the subject? | <input type="checkbox"/> | As an instructor in class? | <input type="checkbox"/> | |
| For advanced knowledge of the subject? | <input type="checkbox"/> | As a student in class? | <input type="checkbox"/> | |
| To learn about operating procedures? | <input type="checkbox"/> | As a reference manual? | <input type="checkbox"/> | |

Your comments:

If you would like a reply, please supply your name and address on the reverse side of this form.

Thank you for your cooperation. No postage stamp necessary if mailed in the U.S.A. (Elsewhere, an IBM office or representative will be happy to forward your comments.)

Reader's Comment Form

IBM Virtual Machine/System Product Operating Systems in a Virtual Machine
Cut or Fold Along Line
Printed in U.S.A.

Fold and Tape

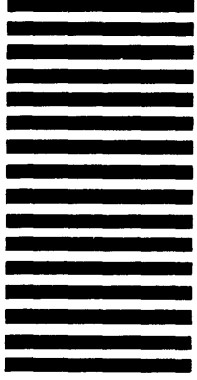
Please Do Not Staple

Fold and Tape



NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL
FIRST CLASS PERMIT NO. 40 ARMONK, N.Y.



POSTAGE WILL BE PAID BY ADDRESSEE:

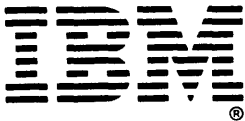
International Business Machines Corporation
Department G60
P. O. Box 6
Endicott, New York 13760

Fold

Fold

If you would like a reply, *please print*:

Your Name _____
Company Name _____ Department _____
Street Address _____
City _____
State _____ Zip Code _____
IBM Branch Office serving you _____



International Business Machines Corporation
Data Processing Division
1133 Westchester Avenue, White Plains, N. Y. 10604

IBM World Trade Americas/Far East Corporation
Town of Mount Pleasant, Route 9, North Tarrytown, N. Y., U. S. A. 10591

IBM World Trade Europe/Middle East/Africa Corporation
360 Hamilton Avenue, White Plains, N. Y., U. S. A. 10601



International Business Machines Corporation
Data Processing Division
1133 Westchester Avenue, White Plains, N.Y. 10604

IBM World Trade Americas/Far East Corporation
Town of Mount Pleasant, Route 9, North Tarrytown, N.Y., U.S.A. 10591

IBM World Trade Europe/Middle East/Africa Corporation
360 Hamilton Avenue, White Plains, N.Y., U.S.A. 10601