GC20-1800-0

Systems

IBM Virtual Machine Facility/370: Introduction

IBM

GC20-1800-0

Systems

IBM Virtual Machine Facility/370: Introduction

The IBM Virtual Machine Facility/370 (VM/370) is a System Control Program (SCP) that has been designed specifically for the IBM System/370. VM/370 manages the IBM System/370 in such a way that multiple remote terminal users appear to have a dedicated computing system at their disposal. Within this "virtual machine" the user may run the operating system of his choice, subject to the restrictions noted in "Appendix C: VM/370 Restrictions" of this manual. The design of VM/370 is based on the IBM Control Program-67/Cambridge Monitor System (CP-67/CMS) which is executed on an IBM System/360 Model 67.

The Conversational Monitor System (CMS) is the major subsystem of VM/370. CMS provides problem solving and program development services to the user, as well as supporting facilities for a remote user who chooses to run some other operating system in his virtual machine.

This manual provides introductory information about the facilities provided by VM/370, and defines the minimum equipment configuration necessary for execution.

IBM

Preface

This manual provides introductory information on the IBM Virtual Machine Facility/370 (VM/370) and its associated subsystem, the Conversational Monitor System (CMS), as well as an overview of the purpose and functions of VM/370.

It is assumed that the user has a prior knowledge of virtual storage concepts as implemented on the IBM System/370 via dynamic address translation. The reader is referred to Part I of the student text publication *Introduction to Virtual Storage in System/370,* Order No. GR20-4260, for an introduction to the concepts and advantages of virtual storage.

A basic knowledge of data processing systems, such as that given in the *Introduction to IBM Data Processing Systems,* Order No. GC20-1684, is assumed. More detailed information about System/370 is available in the publication *IBM System/370 Principles of Operation,* Order No. GA22-7000, and the associated publication *IBM System/360 Principles of Operation,* Order No. GA22-6821. For more information about any individual System/370 model, see the functional characteristics manual and the guide manual for that model.

When the term 3330 is used in this publication, it refers to the IBM 3330 Disk Storage, Model 1 or the IBM 3333 Disk Storage, Model 1.

First Edition (July 1972)

Changes are periodically made to the specifications herein; before using this publication in connection with the operation of IBM systems, refer to the latest IBM System/360 and System/370 SRL Newsletter, Order No. GN20-3060, for the editions that are applicable and current.

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, address comments to: IBM Corporation, Programming Publications Department, 545 Technology Square, Cambridge, Mass. 02139.

© Copyright International Business Machines Corporation 1972

Contents

INTRODUCTION 5

THE VIRTUAL MACHINE CONCEPT 6
The Virtual Machine 6
Virtual Machine Applications 8
Program Maintenance 8
Time Sharing 8
Program Development 8
Backup Systems 9

CONTROL PROGRAM DESCRIPTION 10
Virtual Machine Time Management 10
Virtual Machine Storage Management 10
Virtual Machine I/O Management 11
Spooling Functions 13
Console Functions 14

VIRTUAL MACHINE OPERATING SYSTEMS 15

CONVERSATIONAL MONITOR SYSTEM. 17
CMS Command Language 17
The File System 17
Program Development and Execution 19
Additional Facilities 20

RELIABILITY, AVAILABILITY, AND SERVICEABILITY 21
System/370 Recovery Features 21
VM/370 Recovery Features 21
Recovery Management Support (RMS) 21
Error Recovery Procedures 22
Recording Facilities 22
VM/370 Repair Facilities 22
VM/370 Restart Facilities 22

SYSTEM REQUIREMENTS 24
Machine Requirements 24
CPU's 24
System Consoles 24
Direct Access Storage Devices 24
Direct Access Control Units 24
Magnetic Tapes 24
Magnetic Tape Control Units 24
Printers 24
Readers/Punches 24
Unit Record Control Units 24
Telecommunications 24
Minimum VM/370 Configurations 25
VM/370 Programming Characteristics 25
Maintenance Considerations 25

APPENDIX A. IBM PROGRAMS EXECUTABLE UNDER CMS 27

APPENDIX B. SYSTEM COMMANDS 29
VM/370 Console Functions 29
  General Users 29
  Other Users 30
CMS Commands 30
  File Manipulation Commands 30
  Program Compilation and Execution Commands 30
  Control Commands 30
  Other Commands 31

APPENDIX C. VM/370 RESTRICTIONS 33

APPENDIX D. VM/370 COMPATIBILITY 35

GLOSSARY 37

INDEX 39

FIGURES

Figure 1. Logging into VM/370 and Invoking CMS 6
Figure 2. Multiple Virtual Machines 6
Figure 3. Minidisk Partitioning 7
Figure 4. Virtual Devices: Teleprocessing Control Units 7
Figure 5. Virtual Machines for Concurrent Program Maintenance 8
Figure 6. NORMAL and VIRTUAL=REAL Virtual Machines 12
Figure 7. Virtual Machine Operating Systems 15
Figure 8. A Directory Entry for a CMS Machine 18
Figure 9. CMS File Formats 19

Introduction

The IBM Virtual Machine Facility/370 (VM/370) is a control program that manages the resources of a single computer such that multiple computing systems appear to exist. Each one of these multiple computing systems is called a "virtual" computing system. Each virtual computing system, or virtual machine, is the functional equivalent of an IBM System/370.

A virtual machine is configured by recording appropriate information in the directory file of VM/370. The virtual machine configuration includes counterparts to the components of a real IBM System/370: a virtual operator's console, virtual storage, a virtual CPU, and virtual channels and input/output devices. It is the function of VM/370 to make these components appear real to whichever operating system is controlling the work flow of the virtual machine.

The virtual machines operate concurrently via techniques of multiprogramming. Since the programs executing in any one virtual machine seldom utilize all the resources of an IBM System/370, concurrent execution of virtual machines provides increased utilization of the real computing system. VM/370 overlaps the idle time of one virtual machine with execution in another using techniques similar to those used when multiple job streams are processed by the IBM Operating System (OS) or Disk Operating System (DOS).

The work to be done by the virtual machine is scheduled and controlled by some System/360 or System/370 operating system. The concurrent execution of multiple virtual machines is managed by the control program of VM/370.

An example of a virtual machine operating system is the Conversational Monitor System (CMS), which has been specifically designed to run in a virtual machine under the control of VM/370. CMS essentially is a time-shared subsystem that depends upon VM/370 for real computer management. CMS provides, at a remote terminal, a full range of conversational capabilities: creation and management of files; compilation, testing, and execution of problem programs; and execution of application programs.

Another example of a virtual machine operating system is DOS. DOS would control, as in a real machine, whatever storage was defined in the virtual machine configuration; that storage would in reality be virtual storage managed by VM/370.

VM/370 has been designed for IBM System/370 machines operating in extended control mode and using dynamic address translation. It is a functional extension of CP-67/CMS which has been used since 1968 on the IBM System/360, Model 67. VM/370, like its predecessor, provides: (1) virtual machines and virtual storage, (2) the ability to run multiple operating systems concurrently, and (3) a conversational time-sharing system. Additionally, VM/370 includes support for such devices as the IBM 3330 Disk Storage, Model 1, IBM 3333 Disk Storage, Model 1, and IBM 2305 Fixed Head Storage and offers several performance options that can be used to improve performance in the virtual machine environment.

This publication presents the concepts and facilities of VM/370. It describes virtual machines and their applications, the important characteristics of the control program, and guidelines on acceptable types of virtual machine operating systems. It describes the facilities available with CMS and their application, and that system's use of VM/370 and the environment of virtual machines. Reliability, availability, and serviceability characteristics of VM/370 are discussed, and the equipment requirements of the system are given.

THE VIRTUAL MACHINE

The virtual machine, as implemented by VM/370, is the functional equivalent of an IBM System/370 and its associated devices. The operation of a virtual machine is controlled from a remote terminal such as an IBM 2741 Communications Terminal.

The remote terminal is used initially to access VM/370. A user identifies himself to VM/370 with a pre-established user identification (userid). The userid is associated with the definition of a particular virtual machine configuration by an entry in the VM/370 directory. The user must also enter the correct password for the userid; during this time, printing is inhibited or overprinting occurs as an aid to maintaining the security of the password.

Once VM/370 has been properly accessed, the remote terminal provides the user with the facilities equivalent to those of the operator's console of his virtual machine. The terminal device acts as the operator's console, and VM/370 provides a set of commands to simulate the System Control Panel of the virtual machine. These commands, called console functions, are available to the user at any time. Figure 1 illustrates the activation of a virtual machine, including the loading of a suitable operating system. Further terminal communication would be under control of that system—in this case the CMS subsystem—until VM/370 mode is entered again.

Figure 2 is an example of the types of virtual machines that might reasonably run concurrently on an IBM System/370, Model 145, having 256K of real storage. One machine is assigned to run production work in a batch environment under control of the IBM Disk Operating System (DOS). A second DOS-controlled machine is shown to be running certain programs dependent on a back release of that operating system. Five other virtual machines are shown, each running the Conversational Monitor System for separate users. VM/370 is thus shown to be not only a time-sharing system, but also a system which, through its virtual machines, provides the capability of running production work under differing operating system configurations.

```
vm/370 online
login trauner
ENTER PASSWORD:
(Note: Printing of password inhibited for security)

LOGON AT 11:03:18 EDT WEDNESDAY 05/31/71

ipl cms
```

Figure 1. Logging into VM/370 and Invoking CMS

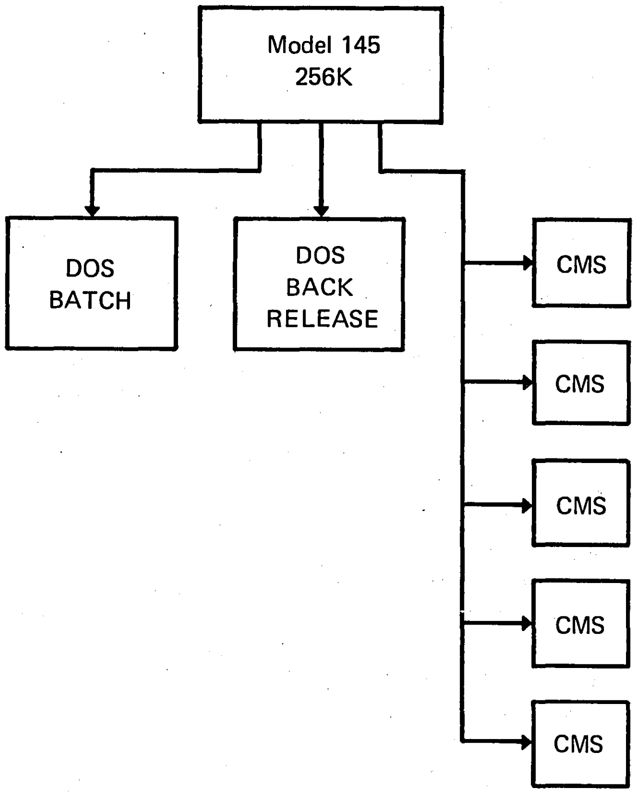

Figure 2. Multiple Virtual Machines

Each virtual machine includes virtual storage, a virtual CPU, and virtual channels and devices in addition to a virtual operator's console. Basic definitions for these components follow:

Virtual Storage—The storage size of the virtual machine is defined in its directory entry in the Virtual Machine Facility (VM/370) and may differ among virtual machines. The storage size can be from 8 thousand to 16 million bytes. This storage appears as real storage to whatever program executes in the virtual machine. VM/370 manages the storage resource as virtual storage, utilizing Dynamic Address Translation on the IBM System/370; this virtual storage management is transparent to the virtual machine. For example, three virtual machines of 256K each could be run on a single

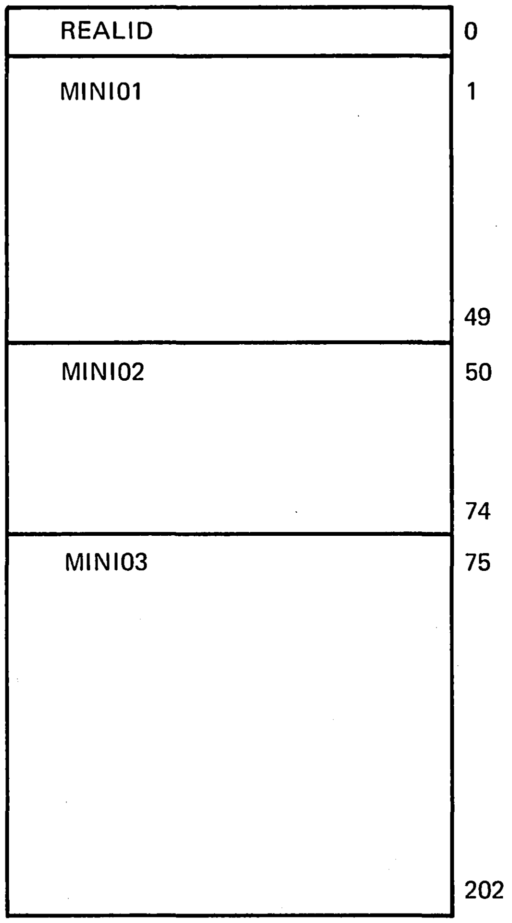

Figure 3. Minidisk Partitioning

real computing system having 384K of real storage. Because the storage appears real to the virtual machine, operating systems such as DOS or OS/MVT can be run without change under control of VM/370 and can, thereby, utilize the facilities of virtual storage. When operating systems such as OS/VS1 (which themselves utilize the Dynamic Address Translation facility of an IBM System/370) are run in a virtual machine, the storage of the virtual machine is itself controlled so as to provide virtual storage.

Virtual CPU–VM/370 provides CPU resources to each active virtual machine through time slicing techniques. The virtual CPU can be run in either basic or extended control mode; the virtual machine operating system can be single task or multitask. Thus, OS/MFT and VS1 as well as CMS and VM/370 can all run in virtual machines. Any IBM System/370 instruction can be executed by the virtual machine except READ DIRECT and WRITE DIRECT. The DIAGNOSE instruction is reserved for special program communication with VM/370.

Virtual Channels and I/O Devices–Devices which are part of a virtual machine configuration are logically controlled by the virtual machine and not by VM/370. Input/Output (I/O) operations, and any error recovery processing, are normally the complete responsibility of the virtual machine operating system. VM/370 converts virtual channel and device addresses to real channel and device equivalents and performs whatever data address translations are necessary. Figure 3 illustrates a real 2319 disk pack which has been mapped as three virtual disks. These disks may belong to three virtual machine configurations or be three logical disks belonging to one virtual machine. The virtual disks are called minidisks since, in this case, they are not mapped to full-sized real equivalents. Service programs distributed with VM/370 create and maintain these minidisks which can be used as normal disk volumes by CMS, DOS, and OS. A subset of the lines of a real transmission control unit (TCU) may be dynamically defined as a virtual TCU for a virtual machine, as shown in Figure 4. A virtual channel-to-channel adapter can be defined either with or without a real equivalent; the former permits a virtual machine to com-

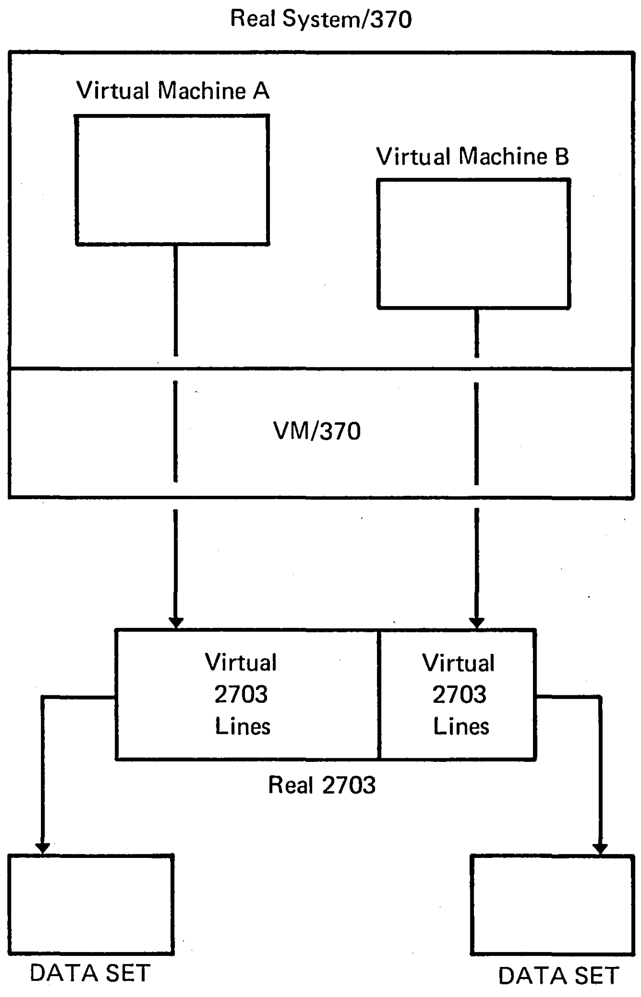

Figure 4. Virtual Devices: Teleprocessing Control Units

municate with a real computing system while the latter allows direct communication between virtual machines.

The programs to be run in a virtual machine should not contain dynamically modified channel programs. (A dynamically modified channel program is one changed between the time the START I/O instruction is begun and the end of the input/output operation, either by the CPU or by the channel program itself.) Some of the channel programs generated by the OS Indexed Sequential Access Method are dynamically modified, but can be translated correctly by VM/370. Other types of dynamically modified channel programs will not execute successfully in the virtual machine environment, unless the appropriate performance option, which eliminates the restriction, is assigned. Refer to "Appendix C: VM/370 Restrictions" for a comprehensive set of VM/370 restrictions, and to the section "Control Program Description" for a discussion of the VM/370 performance options.

The environment created by VM/370 is essentially one of multiple systems, each isolated from the other. Storage is protected because each virtual machine has its own virtual storage; any storage address generated by a virtual machine can refer only to its own virtual storage. VM/370 prohibits a user from accessing virtual input/output devices unless these have been defined as part of the virtual machine configuration or as available for sharing by it and other virtual machines.

VIRTUAL MACHINE APPLICATIONS

The virtual machine as defined in the preceding paragraphs is unique to VM/370. The multiple systems environment created by VM/370 permits the concurrent operation of multiple operating systems. The operating systems may be the same, as when running one copy for programming systems maintenance and a second for batch production, or may be different. The difference may be in the type, level, or configuration.

The availability of concurrent systems under VM/370 permits an installation to perform maintenance and program development functions without the requirement of a dedicated machine. Other applications include the addition of time sharing capabilities to an existing batch operation, and increased flexibilities in providing backup.

Program Maintenance

Modifications and extensions to problem programs or supervisory functions can be made and completely tested in a virtual machine environment. For example, a program temporary fix (PTF) applied to a modifiable copy of an IBM operating system pack can be tested concurrently with the production execution of that same operating system in another virtual machine, provided sufficient direct access storage resources are available. The virtual machine test will be analogous to one made on a real machine providing: (1) there are no time dependencies, (2) dynamically modified channel programs are not used except as noted in "Appendix C: VM/370 Restrictions," and (3) the test is not measuring time.

A possible combination of virtual machines in a VM/370 configuration is shown in Figure 5. Operating system testing is shown running concurrently with batch work and a variety of conversational applications.

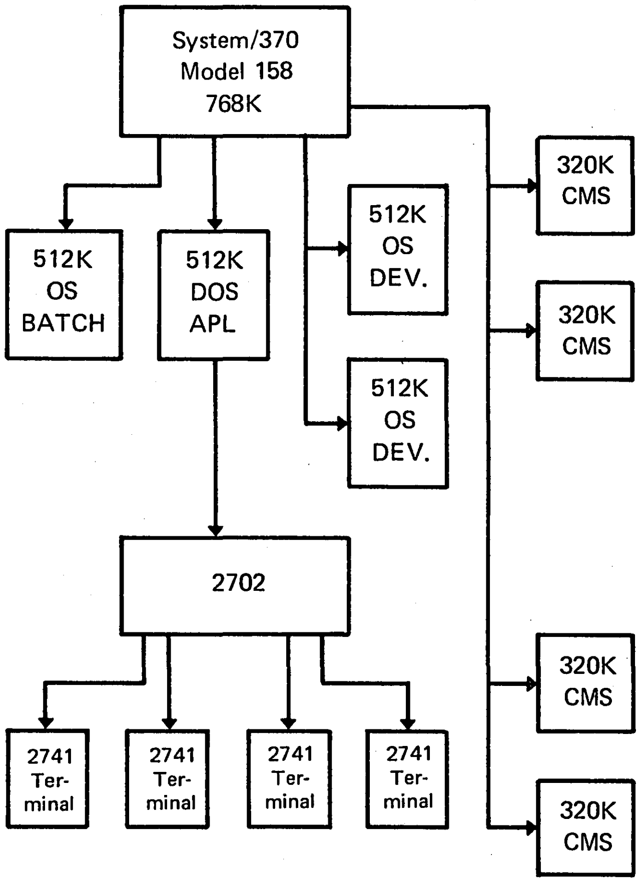

Figure 5. Virtual Machines for Concurrent Program Maintenance

Time Sharing

The Conversational Monitor System is a single-user subsystem designed specifically to run in a virtual machine environment. A time-sharing environment is created when multiple virtual machines are created by VM/370 each controlled by a copy of CMS. These systems operate concurrently with each other as well as with other conversational or batch systems.

Program Development

VM/370 is a terminal-oriented system where each remote terminal is, in effect, the operator's console of a virtual machine. Debugging aids are available at the terminal which parallel the functions to be found at the operator's console, for example, displaying and storing into the general or floating-point registers or into virtual storage, instruction address stopping, and altering the normal flow of execution.

The virtual devices and virtual storage of a virtual machine provide a desirable system independence for program test. The program being developed is not constrained by any fixed, real storage size. The flexibility of virtual device definition for program test has been partially illustrated in Figure 4. Subject to available resources, a virtual machine can be made active whenever it will be useful, thus relaxing normally tight or inflexible testing schedules.

System/360 and System/370 programs can be developed under VM/370. Problem programs using OS macros can be compiled under control of CMS; within certain restrictions these programs may also be tested under CMS. (Refer to the "Program Development and Execution" section in the "Conversational Monitor System" for a more complete discussion of program execution under CMS.) DOS assembler language programs can be compiled under CMS if the installation adds the appropriate DOS macros to the CMS system. Problem programs using DOS macros can be conversationally developed using a "CMS flip-flop" technique. With this technique, program creation and modifications are done at the remote terminal under control of the Conversational Monitor System of VM/370, and actual compilation and test are done when the control of the virtual machine has been "flipped" to DOS. The user specifies which operating system is to control his virtual machine by means of the IPL command of VM/370.

Backup Systems

Virtual device definitions and virtual storage provide flexibility in establishing backup configurations. The storage of the virtual machine may be defined to any necessary size from 8 thousand to a maximum of 16 million bytes. Virtual devices must have real counterparts, but not necessarily a one-for-one correspondence. Two exceptions exist: (1) virtual unit record input/output may be spooled to disk, and (2) a 2311 virtual disk may be mapped to either the top or bottom half of a 2314 or 2319 disk. Virtual device addresses may be assigned without regard to the addresses of the real devices.

Thus, a machine with four channels may be used to back up one having six channels with no change to the original job streams, providing sufficient devices are available. A minidisk may be used as backup to an OS scratch pack which stores only temporary data sets if the total space requirements of the scratch pack are less than a full real disk. It is generally possible to assign more than one such minidisk to a single real disk device, thus reducing the total device requirements of the backup system. The spooling facilities of VM/370 permit a real machine with only a few unit record devices to back up another computer with many unit record devices.

The control program of VM/370 creates and controls virtual machines. Programs will execute on a System/370 virtual machine in a manner which produces output identical to that obtained by execution on a real machine.

A virtual machine is created for a user when he logs into VM/370, on the basis of information stored in the directory file. The entry for each user identification includes a list of the virtual input/output devices associated with the particular virtual machine and the real device mappings.

Additional information about the virtual machine is maintained in the directory file. Included are the VM/370 command privilege class, accounting data, normal and maximum virtual storage sizes, and optional virtual machine characteristics such as extended control mode.

VM/370 controls the execution of virtual machines by (1) permitting only problem state execution except in its own routines, and (2) receiving control after all real computing system interrupts. VM/370 intercepts each privileged instruction and simulates it if the current program status word of the issuing virtual machine indicates a virtual supervisor state; if the virtual machine is executing in virtual problem state, the attempt to execute the privileged instruction is reflected back to the virtual machine as a program interrupt. All virtual machine interrupts (including those caused by attempting privileged instructions) are first handled by VM/370, and are reflected to the virtual machine if an analogous interrupt would have occurred on a real machine.

VIRTUAL MACHINE TIME MANAGEMENT

The real CPU is time sliced to provide the effect of multiple virtual CPUs. Virtual machines, which are executing in a conversational manner, are given access to the real CPU more frequently than those that are not; these conversational machines are assigned the smaller of two possible time slices. VM/370 determines execution characteristics of a virtual machine at the end of each time slice on the basis of the recent frequency of its console requests or terminal interrupts. The virtual machine is queued for subsequent CPU utilization according to whether it is a conversational or nonconversational user of system resources.

A virtual machine can gain control of the CPU only if it is considered dispatchable by VM/370, that is, not waiting for some activity or resource. The virtual machine itself may enter a virtual wait state after an input/output operation has begun. VM/370 also considers the virtual machine not dispatchable if: (1) it is waiting for a page of storage, (2) it is waiting for an input/output operation to be translated and started, or (3) it is waiting for a VM/370 command to finish execution.

A virtual machine can be assigned a priority of execution. This value, taken with run-time characteristics such as the number of pages required for execution and the relative amount of time in execution, influences the dispatching algorithm of VM/370 and, thus, helps to determine the amount of real CPU time made available to the virtual machine. Basically, priority is a parameter affecting the execution of a particular virtual machine as compared with other virtual machines that have the same general execution characteristics. Priority may be assigned by the real machine operator but is more frequently a parameter of the virtual machine's directory entry.

A virtual machine may be assigned the favored execution option by the real machine operator. The effects of the time-slicing algorithm and of any priority value assignments are overridden when this option is in effect. Further, one machine can be allowed access to the real CPU resource a certain percentage of the time; the percentage value is a parameter of the SET FAVORED operator command. This as well as other performance options are discussed in the section, "Virtual Machine Operating Systems."

VIRTUAL MACHINE STORAGE MANAGEMENT

The normal and maximum storage sizes of a virtual machine are defined as part of the virtual machine configuration in the VM/370 directory. The user may temporarily redefine his virtual storage size to any value that is a multiple of 4K and not greater than his maximum defined value. The minimum and maximum storage sizes are 8K and 16M bytes, respectively. VM/370 implements this storage as virtual storage. The storage may appear as paged or nonpaged to the virtual machine, depending upon whether the extended control mode option has been specified for that virtual machine. This option is required if operating systems that control virtual storage, such as OS/VS1 or VM/370, are to be run in the virtual machine.

Storage in the virtual machine is logically divided into 4096 byte areas called pages. A complete set of segment and page tables is used to describe the storage of each virtual machine. These tables are maintained by VM/370 and reflect the allocation of virtual storage pages to blocks of real storage. Virtual storage addressing is accomplished through use of these tables by the System/370 machine. Storage in the real machine is logically and physically divided into 4096 byte areas called page frames or blocks.

Only referenced virtual storage pages are kept in real storage, thus optimizing real storage utilization. Further, a page can be brought into any available page frame; the necessary relocation is done during program execution by a combination of VM/370 and dynamic address translation on the System/370. The active pages from all logged-in virtual machines and from the pageable routines of VM/370 compete for available page frames. When the number of page frames available for allocation falls below a threshold value, VM/370 determines which virtual storage pages currently allocated to real storage are relatively inactive and initiates suitable page-out operations for them.

Inactive pages are maintained on a direct access storage device. If an inactive page has been changed at some time during virtual machine execution, VM/370 assigns it to a paging device, selecting the fastest such device with available space. If the page has not changed, it remains allocated in its original direct access location and is paged into real storage from there the next time the virtual machine references that page. A virtual machine program can use the DIAGNOSE instruction to communicate to VM/370 that the information from a specific page(s) of virtual storage is no longer needed; VM/370 then releases the areas of the paging device(s) which had been assigned to hold the specified page(s).

Paging is done on demand by VM/370. This means that a page of virtual storage is not read (paged) from the paging device to a real storage block until it is actually needed for virtual machine execution. No attempt is made by VM/370 to anticipate what pages might be required by a virtual machine. While a paging operation is being performed for one virtual machine, another virtual machine can be executing. Any paging operation initiated by VM/370 is transparent to the virtual machine.

If the virtual machine is executing in extended control mode with translate on, then two additional sets of segment and page tables are maintained. The virtual machine operating system is responsible for mapping the virtual storage created by it to the storage of the virtual machine. VM/370 uses this set of tables in conjunction with the page and segment tables created for the virtual machine at login time to build shadow page tables for the virtual machine. These shadow tables map the virtual storage created by the virtual machine operating system to the storage of the real computing system. The tables created by the virtual machine operating system may describe any page and segment size permissible in the IBM System/370.

The system operator may assign the reserved page frames option to a single virtual machine. This option, specified by the SET RESERVE command, assigns a specific amount of the storage of the real machine to the virtual machine. VM/370 will dynamically build up a set of reserved real storage page frames for this virtual machine during its execution until the maximum number "reserved" has been reached. Pages for the selected virtual machine are paged into and out of page frames from the reserved set if possible. Since other virtual machines' pages are not allocated from this reserved set, the effect will be that the most active pages of the selected virtual machine will remain in real storage.

During the process of VM/370 system generation, the installation may specify that a single virtual machine is to be given an option called virtual=real. With this option, the virtual machine's storage is allocated directly from real storage at the time VM/370 is initially loaded, and remains so allocated unless released via operator command. All pages except page zero are allocated to the corresponding real storage locations. Refer to "Appendix C: VM/370 Restrictions" for a discussion of the implications of using this option in a production environment. In order to control the real computing system, real page zero must be controlled by VM/370. Consequently, the real storage size must be large enough to accommodate the VM/370 nucleus, the entire Virtual=Real virtual machine, and the remaining pageable storage requirements of VM/370 and the other virtual machines.

The virtual=real option will improve performance in the selected virtual machine since it removes the need for VM/370 to perform paging operations for the selected virtual machine. Performance in other virtual machines may be adversely affected unless enough real storage is available for their paging requirements. The virtual=real option is necessary whenever programs that contain dynamically modified channel programs (excepting those of OS ISAM) are to execute under control of VM/370. Figure 6 illustrates real storage allocation for both a normal DOS batch virtual machine, and one described with the virtual=real option.

VIRTUAL MACHINE I/O MANAGEMENT

A virtual disk device can be shared among multiple virtual machines. Virtual device sharing is specified in the directory entry or by a user command. If it is the latter, the user must supply an appropriate password to the system before gaining access to the virtual device. A particular virtual machine may be assigned read-only or read-write access to a shared disk device. VM/370 verifies each virtual machine input/output operation against the parameters in the virtual machine configuration to ensure device integrity.

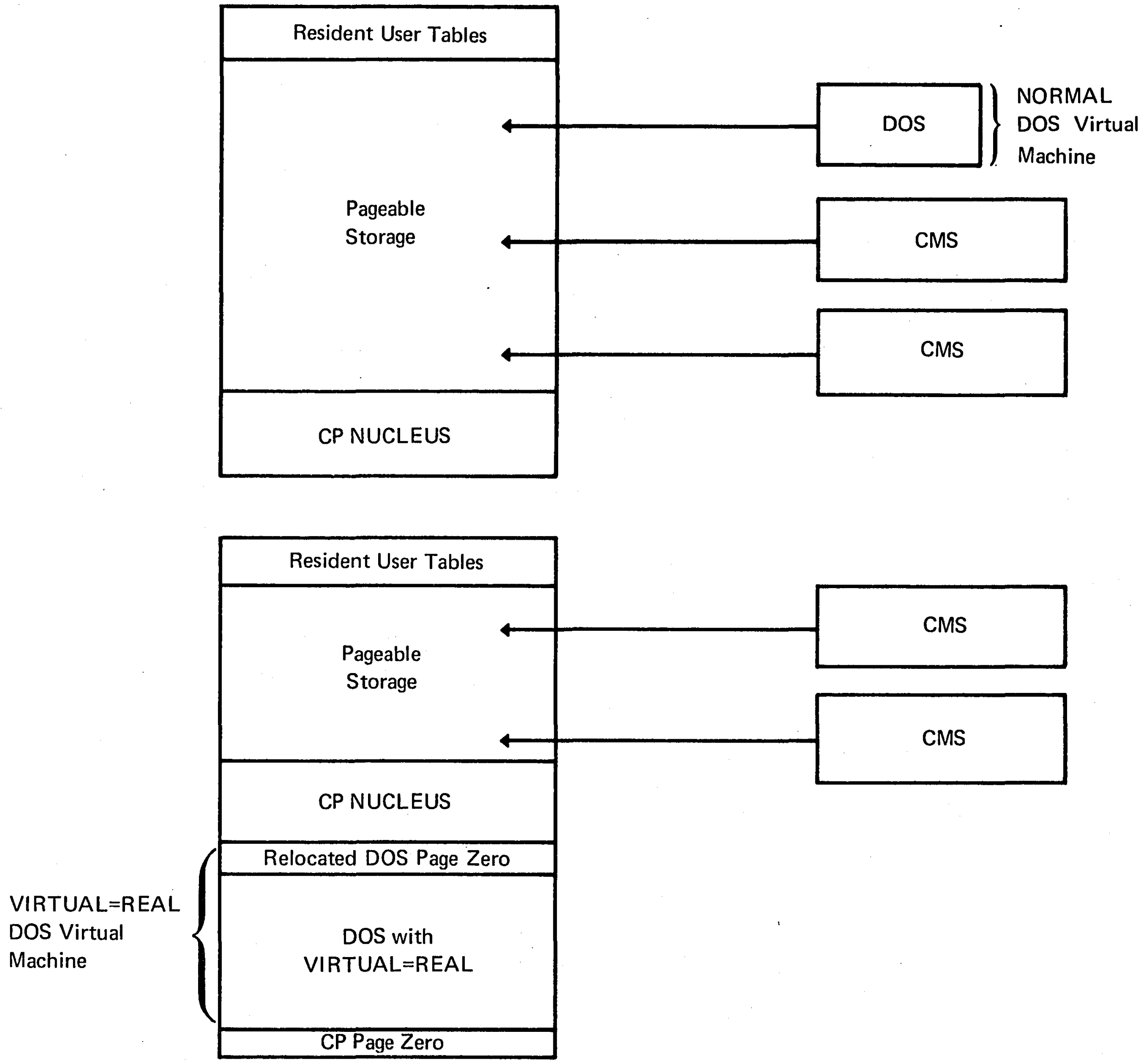

Figure 6. NORMAL and VIRTUAL=REAL Virtual Machines

Virtual disk devices may be defined for temporary use by a virtual machine. In that case, VM/370 allocates real disk storage to the virtual machine only while the virtual machine is active.

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 directory entry of the virtual machine, or they may be attached to (or detached from) the virtual machine's configuration while it remains logged on. Virtual devices may be dedicated, as when mapped to a fully equivalent real device; shared, as when mapped to a minidisk or when specified as a shared virtual device; or spooled by VM/370 to intermediate direct access storage.

In a real machine running under control of OS, input/output operations are normally initiated when a problem program requests OS to issue a START I/O instruction to a specific device. Device error recovery is handled by the operating system. In a virtual machine, OS can perform these same functions, but the device address specified and the storage locations referenced will both be virtual. It is the responsibility of VM/370 to translate the virtual specifications to real.

Because supervisor state in a virtual machine is itself virtual, VM/370 will gain control when the START I/O instruction is issued by the virtual machine operating system. VM/370 will copy into its own work area the channel command list specified by the operating system, and page into real storage all virtual storage locations required for data transfer. The specified pages are fixed in real storage until completion of the input/output operation. If a single channel command word has specified a data area extending over multiple pages of contiguous virtual storage, VM/370 will generate channel programs which utilize channel indirect data addressing to handle discontiguous page frames. If the virtual device is a minidisk, any cylinder numbers specified are modified to reflect the true location of the data. The virtual device address is mapped to the real device and an actual input/output operation scheduled. During this processing, the virtual machine has been marked as not executable by VM/370. When the virtual machine gains control, it is given a suitable condition code (as on a real machine) to indicate the status of the START I/O operation.

In addition, the interrupts caused by the input/output operation will be reflected to the virtual machine for its interpretation and processing. If input/output errors occur, VM/370 will not initiate error recovery operations; these are the responsibility of the virtual machine operating system. Basic error recording is provided by VM/370.

A virtual machine assigned the dedicated channel option has that channel and all of its devices for dedicated use. VM/370 will translate the virtual storage locations specified in channel commands to real locations as well as perform any necessary paging operations, but will not perform any device address translations. The dedicated channel option allows VM/370 to bypass its own channel scheduling and interrupt reflection routines for all the devices on the channel. The virtual devices on a dedicated channel must have direct, real equivalents: for example, minidisks are not allowed, and the virtual and real device addresses must be identical. A channel dedicated to a virtual machine cannot be used by VM/370 or by any other virtual machine. Virtual machines may have a mixture of dedicated and nondedicated channels.

The initiation of a virtual input/output operation by VM/370 can be simplified through use of the DIAGNOSE interface by the virtual machine operating system. The Conversational Monitor System, which has been designed specifically for the virtual machine environment, uses this interface instead of the normal START I/O instruction for most of its input/output operations. When DIAGNOSE is used, VM/370 assumes responsibility for input/output error recovery operations.

Input/output operations initiated by VM/370 for its own purposes, for example, paging and spooling, are performed directly and are not subject to the translation process described in the preceding paragraphs.

SPOOLING FUNCTIONS

A virtual unit record device, which is mapped directly to a real unit record device, is said to be dedicated. The real device is then controlled completely by the virtual machine's operating system.

VM/370 facilities exist which allow multiple virtual machines to share unit record devices. Since virtual machines controlled by CMS ordinarily have modest requirements for unit record input/output such device sharing is quite advantageous, and it is the standard mode of system operation.

VM/370 controls and schedules the operation of the real unit record devices via spooling techniques as described below. Virtual machine START I/O instructions directed to those unit record devices designated as SPOOL devices in the directory entry are intercepted and modified by VM/370. VM/370 generates another input/output operation, transparent to the virtual machine, which replaces the one specified. The new operation is directed to a VM/370 disk area that acts as intermediate storage between the real unit record device and the virtual machine. The data transfer operation between a SPOOL unit record device and the virtual machine is in reality between a VM/370 spool file and the virtual machine. Spool file records are page size (that is, 4096 byte blocks) and are transferred between storage media via the VM/370 paging mechanism.

Input spool files, that is, data available at a virtual card reader, can be created when the real machine operator feeds cards via the real card reader which have been preceded by a record specifying the appropriate virtual machine userid. A second method of creating input spool files is via an IBM 2780 Data Transmission Terminal. VM/370 provides a virtual machine monitor program which transfers files between a 2780 and the devices of any specified virtual machine; cards at a remote location can be made available to any virtual machine.

Output spool files are also on direct access storage and are created automatically when the virtual machine operating system writes to a virtual punch or printer. Real output is scheduled for a real printer or punch, or for remote output, whenever a user logs off the system or issues a VM/370 spooling command to close the file.

Spooling operations will cease if the direct access storage space assigned to spooling has been exhausted, and the virtual unit record devices will appear in a not ready status. The system operator may make additional spooling space available by purging existing spool files or by assigning additional direct access storage space to the spooling function.

Specific files can be transferred from the spooled card punch or printer of a virtual machine to the card reader of the same or another virtual machine. (A virtual card reader is not limited to 80 characters.) Files transferred between virtual unit record devices by the spooling routines are not physically punched or printed. With this method, files can be made available to multiple virtual machines, or to different operating systems executing at different times in the same virtual machine.

VM/370 spooling includes many desirable options for the virtual machine user and the real machine operator. These include printing multiple copies of a single spool file, backspacing any number of printer pages, and defining spooling classes for the scheduling of real output.

CONSOLE FUNCTIONS

The VM/370 console functions allow the user to control his virtual machine from the terminal, much as an operator controls a real machine. Virtual machine execution can be stopped at any time by use of the terminal's attention key; it can be restarted by typing in the appropriate VM/370 command. The user can simulate external, attention, and device ready interrupts to the virtual machine. Virtual storage and virtual machine registers can be inspected and modified, as can status words such as the PSW and the CSW. Extensive trace facilities are provided for the virtual machine, as well as, effectively, a single-instruction mode. Commands are available to invoke the spooling and disk sharing functions of VM/370.

Console functions are classified into one or more privilege classes. The directory entry for each user assigns one or more privilege classes. The classes are system operator, operator, system programmer, spooling operator, systems analysts, customer engineering, and general users. Commands in the system analysts class may be used to inspect real storage locations, for example, VM/370 data values, but may not be used to make modifications to real storage. Commands in the operator class provide real resource control capabilities. System operator commands include all those relating to virtual machine performance options, such as assigning a set of reserved page frames to a selected virtual machine. (Refer to "Appendix B: System Commands" for specific information on VM/370 system commands.)

To be managed efficiently, a virtual machine should be controlled by a monitor or operating system. The system may be single task, such as CMS or a single partition DOS, or multi-task, as OS/MVT. The system may be multi-access, such as APL\360-DOS (CP option), supporting within its own programs a number of remote terminals. As long as the operating system does not violate any of the restrictions listed in "Appendix C: VM/370 Restrictions," it will execute on a virtual machine in the same manner as on a real machine.

The virtual machine operating system executes in real problem state and without access to real page zero: in this way, VM/370 can intercept all virtual machine privileged instructions and initially process all interrupts. Unless executing in a Virtual=Real virtual machine, the virtual machine operating system is pageable, that is, it is loaded into the storage of the virtual machine and paged into real storage on demand. It competes for real storage with all other virtual machines and with certain portions of VM/370 itself. By contrast, the heavily used portions of VM/370 are not pageable, do not compete with virtual machines for the real storage resource, and create a fixed real storage requirement. The virtual machine operating system gains control of the real CPU by means of the time-slicing algorithm of VM/370. Interaction between this algorithm and an operating system's own multiprogramming technique is complex.

Multi-access operating systems are those that execute in a virtual machine and directly control terminals. These terminals need not be of the type supported by VM/370 as virtual operators' consoles, but may be of any type that can be supported by the virtual machine operating system and the System/370.

System/360 and System/370 operating systems, which are supported under control of VM/370, are shown in Figure 7. VM/370 provides to each of these the flexibilities of virtual device support and virtual storage. The operating systems themselves execute as though they were controlling real devices and real storage.

Operating systems that are using dynamic address translation on the IBM System/370, for example, OS/VS1 or VM/370, can run in a virtual machine environment but will experience performance degradation in addition to that described below because of the double management of storage. VM/370 will control the real storage of the computing system. The virtual machine operating system will control what appears to be real storage, that is, the storage of the virtual machine. If the Virtual=Real option is selected for a virtual machine,

Batch or *Single User Interactive*	*Multi-Access*
DOS	APL\360-DOS (CP option)
DOS/VS	VM/370
OS/PCP	
OS/MFT	*Conversational*
OS/MVT	CMS
OS/VS1	
OS/VS2	
OS–ASP	
PS44	

Figure 7. Virtual Machine Operating Systems

VM/370 need not manage virtual machine storage as extensively as usual since, except for page zero, the virtual machine storage will have the same range of addresses as its allocation to real storage. The installation should be aware of the restriction on input/output operations initiated by a machine with the Virtual=Real option. Refer to "Appendix C: VM/370 Restrictions."

The virtual machine environment of VM/370 offers potential for increased throughput for program development and maintenance projects. Productivity is often closely related to system availability schedules. Under VM/370, testing under a particular operating system can be scheduled easily in a virtual machine; it is unnecessary to reserve the real machine on a dedicated basis (unless the test involves time measurements, or the program cannot execute in a virtual machine). Further, the conversational characteristics of VM/370 improve job turnaround.

The performance of an operating system in a virtual machine is influenced by several factors. The remainder of this section discusses certain performance characteristics and summarizes the performance options of VM/370.

First, performance is related to the overall characteristics and configuration of the VM/370 system, as follows:

- Total number of virtual machines in execution
- Type of work being done by each virtual machine
- Type and capacity of the primary paging devices

- Number of channels available
- Amount of real storage available

Certain characteristics and options of the operating system itself influence its performance in a virtual machine. Since every privileged instruction executed by a virtual machine operating system is intercepted by VM/370, the frequency of their use will adversely affect the operating system performance.

Effective utilization of the storage of a virtual machine is very important and can be significantly affected by installation action. When possible, an operating system such as OS should be generated with its components resident since VM/370 in combination with the dynamic address translation of the IBM System/370 can provide storage management that is automatic and highly efficient. Problem programs should usually allow the paging mechanism of VM/370 to manage storage automatically rather than be structured manually with complicated overlays; this permits simpler program design and easier testing.

One or more segments of virtual storage can be shared among virtual machines. The information to be shared must be read-only; it may be data or reentrant program modules. The information to be shared must be part of a monitor or operating system that has been recorded or "saved" on a VM/370 system disk. Storage segments cannot be shared among virtual machines when executing in extended control mode with translate on.

The performance options of VM/370 can be used to enhance the run characteristics of certain virtual machines. As described in the "Control Program Description," some of these functions can be applied to only one virtual machine at a time. Although each function can be applied to different virtual machines, this would not be normal practice if optimum performance is required. Application of these performance options to a particular virtual machine is said to create a "preferred virtual machine." It is generally true that a virtual machine doing a high number of input/output operations, initiated by normal START I/O instructions, will suffer performance degradation caused by the necessary channel program translations and virtual-to-real device mappings. Two of the VM/370 options directly address this area: (1) dedicated channels and (2) virtual=real. (Refer to "Appendix C: VM/370 Restrictions" for a discussion of the latter performance option in a production environment.) The favored execution option is used to improve the response-time characteristics seen by the users of a multi-access operating system such as APL\360 when run under VM/370. It also can be used to assure a certain percentage of CPU time to a virtual machine running production batch. The reserved page frames option increases the likelihood that a page of virtual storage will, when needed, be already allocated to real storage. It is particularly useful for execution of a multi-access (that is, teleprocessing) operating system, or when running an operating system that supports virtual storage.

The Conversational Monitor System (CMS), the major subsystem of VM/370, provides a comprehensive set of conversational facilities to a single user. CMS has been designed specifically for the virtual machine environment of VM/370 and does not manage such real machine resources as time or storage.

CMS together with VM/370 is a time sharing system suitable for problem solving, program development, and general conversational work. It includes support for several programming languages, file manipulation commands, utilities, and debugging aids. Additionally it provides facilities to simplify the operation of other operating systems in a virtual machine environment when controlled from a remote terminal. In the latter case, CMS capabilities are used to create and modify job streams, and to analyze virtual printer output. A specific example is discussed below, under "Program Development."

CMS is an extension of the Cambridge Monitor System, a major component of CP-67/CMS. (Refer to "Appendix D: VM/370 Compatibility" for specific information concerning the relationships between these systems. Also refer to "Appendix A: IBM Programs Executable Under CMS" for information concerning the ordering procedures for those programs that are described in this section as available under CMS.)

Part of the environment of CMS is related to the virtual machine environment created by VM/370. Each user can be isolated completely from the activities of all other users, and each machine in which CMS executes has virtual storage available to it. The console functions of VM/370 are recognized by CMS, thereby allowing the time sharing user to implicitly alter his command environment. The console functions allow messages to be sent to the operator or to other users, virtual devices to be dynamically detached from the virtual machine configuration, and facilities additional to those provided by CMS to be invoked. "Appendix B: System Commands" provides more complete examples of the system commands available.

CMS COMMAND LANGUAGE

The command language of CMS offers the terminal user a wide range of functions. A variety of programming languages are available for use with CMS, as described in "Appendix A: IBM Programs Executable Under CMS." The languages BASIC, FORTRAN, and PL/I are useful when CMS is being used for problem-solving applications, while COBOL, assembler, and again PL/I are pertinent to commercial program development applications.

The functions available in CMS also include user file manipulation and utility facilities, program execution control, general system control, and debugging facilities. "Appendix B: System Commands" includes a discussion of the CMS commands according to these functional categories.

The command names of CMS may be truncated by the user. The system maintains an ordered list of command names; this list is used to determine which command to invoke when truncation is used. The command list sequence and the valid limit of truncation may be modified by the installation. Each user (or installation) may define synonyms for any or all command names, and may query the current set.

The EXEC processor of CMS can be used to define new commands which are a combination of existing commands. Such new commands, called EXEC procedures, can be set up simply to eliminate the tedious re-keying of frequently used sequences of commands. Logical capabilities exist in the EXEC processor and are invoked with statements analogous to the GOTO, IF, and LOOP statements familiar to high level language users. A special EXEC procedure can be invoked automatically when the user issues his first command in the CMS environment; its purpose is to initialize the system according to that specific user's "profile" record.

In addition, any program on any CMS system or user disk can be invoked by name as a command. The installation, as well as the individual user, has significant flexibility for tailoring the CMS system.

THE FILE SYSTEM

The Conversational Monitor System interfaces with virtual disks, tapes, and unit record equipment. The CMS residence device is maintained as a read-only, shared system disk. Permanent user files may be accessed from up to nine active disks. Logical access to these virtual disks is controlled by CMS while VM/370 facilities manage the device sharing, virtual to real mapping, etc. Figure 8 is a sample VM/370 directory entry describing a virtual machine configured to run CMS. It defines two virtual disks and the required unit record devices. The latter must be set up for VM/370 spooling as is shown here since CMS includes no error recovery procedures for dedicated unit record operations.

The first virtual disk shown in Figure 8 is defined by the LINK entry to share the CMS system disk on a read-only basis; the CMS system disk is owned by the user CMSSYS and is assigned the virtual device address 190 in both the CMSSYS and the TRAUNER virtual machine configurations. The second virtual disk defined in Figure 8 is owned by the user TRAUNER. It has virtual address 191, is a minidisk located on the real volume labeled CMSVL1, and occupies 5 cylinders starting with cylinder 025 on that volume.

```
USER TRAUNER FRANZ
ACCOUNT 5796
CONSOLE 009 3215
SPOOL 00C 2540 READER
SPOOL 00D 2540 PUNCH B
SPOOL 00E 1403 A
LINK CMSSYS 190 190 R
MDISK 191 2314 025 005 CMSVL1 W
```

Figure 8. A Directory Entry for a CMS Machine

User files in CMS are identified with three designators. The first is filename. The second is a filetype designator which may imply specific file characteristics to the CMS file management routines. The third is a filemode designator which describes the location and access mode of the file.

The language processors available for use with CMS assume particular input filetypes, for example, FORTRAN or PLI, but the file manipulation and listing commands do not. Files of a particular filetype form a logical data library for a user, for example, the collection of all FORTRAN source modules, or of all EXEC procedures. This allows selective handling of specific groups of files with minimum input by the user.

Three formats of the file identification are shown in the following example:

```
PROG1
PROG1 COBOL
PROG1 COBOL A1
```

Both the filename and filetype may begin with any alphameric character. If the filemode designator is omitted, the active disks are searched according to a standard, system defined, order of search. A user disk is associated with a mode letter by the ACCESS command when the disk is made active; a maximum of nine user disks may be made active. The filemode consists of a single letter (A-G, S, Y, or Z) followed by a number (0-5). S is used to denote the System disk. The filemode number may indicate that the file is private, read/write, read/only, or in simulated OS-format.

User files can be created directly from the terminal with the EDIT command of CMS. EDIT provides extensive context and line number editing services. File characteristics such as record length and format, tab locations, and a serialization option can be specified. The system includes standard definitions for certain filetypes, among them FORTRAN, ASSEMBLE, PLI, and COBOL.

The editor of CMS is also used to modify existing disk files. Specific records in the file can be located by providing the editor with the contents of the record desired, such as:

```
LOCATE/DATA DIVISION/
```

The file will be scanned by the editor until the first occurrence of the specified characters is reached; the record containing this data is then typed at the terminal. Once located, record contents can be easily changed, as in the sequence which follows:

```
LOCATE /IDEMTIFICATION/
CHANGE /IDEMTIFICATION/IDENTIFICATION/
```

All appearances of a specific string of characters within a file can be modified with a single command.

```
CHANGE /USER IDENTIFICATION/USERID/ * *
```

The above example illustrates such a "global" change with source program records that could be used as input to the American National Standard COBOL compiler.

CMS dynamically allocates compiler work files at the beginning of command execution on whichever active disk has the greatest amount of available space, and de-allocates them at completion. Compiler object decks and listing files are normally allocated on the same disk as the input source file or on the primary read-write disk. They are identified by utilizing the input filename together with the filetypes TEXT and LISTING. The locations of these data files may be overridden by the user.

The size of a single user file is limited to 65,767 records and must be totally contained on a single virtual disk. The file management system limits the number of files on any one virtual disk to 3400. All CMS disk files are written as 800-byte records, chained together by a specific file entry which is stored in a table called the user file directory; a separate user file directory is maintained for, and on, each virtual disk. The data records may be discontiguous, and are allocated and de-allocated automatically. Figure 9 illustrates the file structure used by CMS. The user file directory, or a specified subset of it, is brought into virtual storage when the disk is made available to the CMS user; it is updated on the virtual disk at least once per command if the status of any file on that disk has been changed. CMS automatically opens and closes all accessed files (including spool files) for each command or user program executed. (Refer to

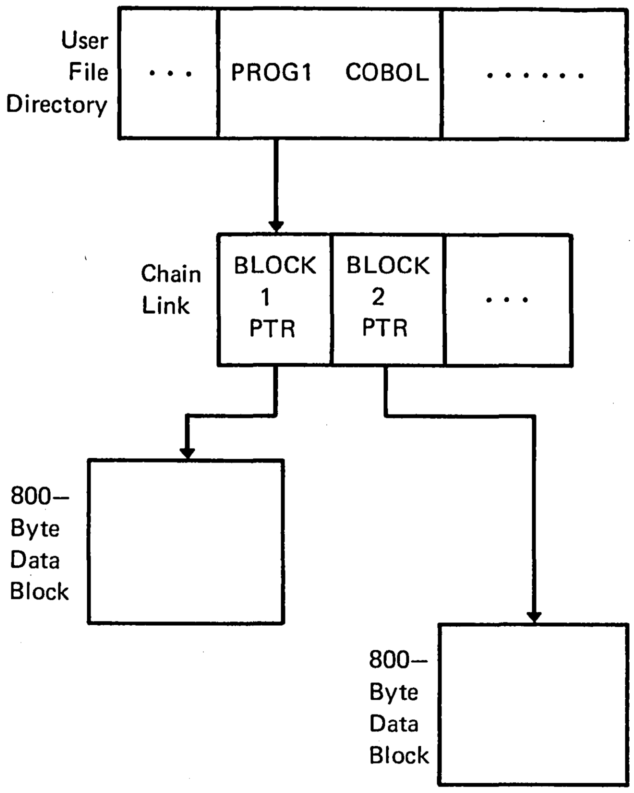

Figure 9. CMS File Formats

"Program Development Facilities" for a description of the relationship between OS problem programs and the CMS file system.)

Virtual disks may be shared by CMS users; the facility is provided by VM/370 to all virtual machines, although a user interface is directly available in CMS commands. Specific files may be spooled between virtual machines to accomplish file transfer between users. Service commands allow such file manipulations as writing from an entire disk or from a specific disk file to a tape, printer, punch, or the terminal. Other commands write from a tape or virtual card reader to disk, rename files, copy files, and erase files. Macro libraries and program libraries are provided by CMS, and special commands are provided to maintain and use them. CMS files can be written onto and restored from unlabeled tapes via CMS service commands.

Problem programs which execute in CMS can create files on unlabeled tape in any record and block size; the record format can be fixed, variable, or undefined.

PROGRAM DEVELOPMENT AND EXECUTION

The Conversational Monitor System includes a wide range of functions to facilitate program development. These include commands to create and compile source programs, to modify and correct source programs, to build test files, to execute test programs and to debug from the terminal. The commands of CMS are especially useful for OS program development, but also may be used in combination with other operating systems to provide a virtual machine program development tool.

The compilers executable under CMS include the VM/370 System Assembler, OS PL/I Optimizing Compiler, OS FORTRAN IV (G1, Code and Go, and H Extended), and OS Full American National Standard COBOL, Version 4. (See "Appendix A: IBM Programs Executable Under CMS" for status information on these and other compilers executable under CMS.) The compilers are invoked within the conversational environment of CMS; the normal mode of execution is to run the compilation to completion, type any diagnostic messages at the terminal, and make the listing file available for inspection at the terminal or for offline printing. Most object programs produced by CMS may be executed under CMS for direct problem solving. Programs which use certain OS system functions, described below, must be written onto an OS-format disk and run under the appropriate operating system.

CMS utilizes the OS compilers via interface modules. To support the compilers, CMS simulates many of the OS macros. The sequential, direct, and partitioned access methods are logically simulated; the data records are physically maintained in the chained 800-byte blocks that are standard to CMS and are processed internally to simulate OS data set characteristics. Several OS Supervisor Call functions including GETMAIN/FREEMAIN and TIME are simulated.

The OS macros that are not simulated include those which support the Indexed Sequential Access Method and the telecommunications access methods. Functions related to multitasking are either ignored by CMS or modified to achieve single task execution. An OS problem program that uses only those functions for which simulation code exists may be tested and run under CMS. For example, all FORTRAN IV (G1) language functions will execute under CMS while COBOL programs that use ISAM will not.

If the program to be tested uses OS functions which are not simulated, or if the program is designed for some other operating system (such as DOS), then a CMS flip-flop technique may be used. In the virtual machine environment, the process of a single user running different operating systems in the same virtual machine is called flip-flop to emphasize the alternation of systems. The virtual machine must be configured to run both CMS and the other operating system.

In the CMS flip-flop technique, the user first loads the Conversational Monitor System into the virtual machine. The editor is used to make any necessary updates to the source program. Spooling facilities are used to copy the program (integrated into a suitable operating system job stream) into the virtual card reader. The user issues the IPL command to load his other operating system (such as DOS) and begin the compilation. When the reader is empty, that is, when the job stream has been processed, CMS is reloaded with the IPL command. The spooled printer output generated by the other operating system may be read onto a CMS user disk, inspected for diagnostic messages, then optionally scheduled for printing. Additional compilations in the development process follow the procedure above. To execute the resultant program under the other operating system, a suitable job stream should be created as a CMS file and transferred to the virtual card reader.

With the CMS flip-flop technique, a user can test programs which use OS ISAM under OS after performing all compilations under CMS.

The debugging facilities of CMS permit a user to set instruction address stops in his program, to examine and modify virtual registers and virtual storage, and to trace all SVC interrupts. User-selected interrupts may be traced with output directed to either a virtual printer or the terminal.

Symbolic debugging capabilities are available to FORTRAN programmers using Code and Go FORTRAN or FORTRAN IV (G1) in the FORTRAN Interactive Debug Program Product.

ADDITIONAL FACILITIES

The Conversational Monitor System includes a batch job facility which accepts input streams in CMS command formats. The CMS Batch facility is automotically reloaded at the start of each new job in the stream to ensure the integrity of the Batch facility and to provide accounting information for each job. Facilities are available to create card or printer output files that may be transferred to the user. Other files created during execution of a job under the CMS Batch facility are automatically erased at its completion.

Programs that are not a part of the Conversational Monitor System may be coded to interface with its file system via CMS or OS macros and, thus, become new system commands. An installation, or a single user, may add to the capabilities of CMS in this way. The following program runs under CMS. Refer to "Appendix A: IBM Programs Executable Under CMS" for a description of the availability and ordering status of this program.

Program Name	*Capability*
BASIC	The BASIC language provides the facility of personal computing for a user at a remote terminal. The BASIC language implemented under CMS is the same language as that available with CALL-OS adapted for use under CMS and is an extension of the language originally developed at Darthmouth College, Hanover, New Hampshire.

The IBM System/370 and its System Control Programs provide an extensive group of advanced reliability, availability, and serviceability (RAS) features. These features address the system requirements demanded by online and time-sharing operations.

The objective of the RAS features of the IBM System/370 is to reduce the frequency and impact of system interruptions caused by machine failure. In many cases, the system can be run in a degraded mode so that maintenance can be deferred to scheduled maintenance periods. When permanent failures do occur, their impact can be reduced by fast isolation and repair of the malfunction. RAS features are as follows:

1. Recovery facilities are provided to reduce the number of failures that cause a complete system termination. This permits deferred maintenance.
2. Repair procedures include online diagnosis and repair of malfunctions concurrent with VM/370 execution. The effect of such repairs on system availability is thus reduced.

SYSTEM/370 RECOVERY FEATURES

The IBM System/370 attempts correction of most machine errors without program assistance. VM/370 is notified, via an interrupt, of both intermittent and permanent machine errors so that error recording and recovery procedures can be initiated.

The following recovery features are implemented in the IBM System/370:

1. CPU retry of failing CPU operations.
2. Validity checking on processor and control storage to correct all single-bit errors.
3. Input/output operation retry facilities including an extended channel status word (ECSW), which provides channel retry data to channel and control unit retry procedures.
4. Expanded machine check interrupt facilities to improve error recording and recovery procedures.

A detailed description of the recovery facilities for any particular IBM System/370 model can be obtained in the systems guide for that CPU, for example, the publication *A Guide to the IBM System/370 Model 145*, GC20-1734.

VM/370 RECOVERY FEATURES

Recovery Management Support (RMS)

The primary objectives of RMS are: (1) to reduce the number of system terminations that result from machine malfunctions and (2) to minimize the impact of such incidents. These objectives are accomplished by programmed recovery which allows system operations to continue whenever possible, and by recording of the system status for both transient (corrected) and permanent (uncorrected) errors. The recovery management functions of VM/370 are provided by a machine check handler and a channel check handler.

Two modes of system operation are possible with VM/370:

- Full recording mode
- Quiet (or nonrecording) mode

In full recording mode, all machine checks cause a machine check interrupt to be taken and logouts to occur. This is the normal mode of operation of VM/370. In quiet mode, all, or certain, soft machine check interrupts are disabled. (Soft machine checks are those from which the CPU has recovered.) Quiet mode is used to permit system operation with limited error recording when a large number of transient errors are occurring. The system operator has the facility to enable or disable all soft recording by use of the SET command.

When a system recovery soft machine check occurs, a recovery record containing pertinent information about the error is written. This record includes the data in the fixed logout area, the date, and the time of day. The operator is informed that a soft machine check has occurred. If a soft machine check occurs during virtual machine execution, it is handled as described above but is not reflected to the virtual machine.

When a permanent machine check occurs during instruction processing, the error is analyzed to determine whether or not it is correctable by programming. System damage, time-of-day clock, and CPU timer errors that result in a machine check interrupt are not correctable; the real computing system is placed in a disabled wait state. Uncorrectable or unretryable CPU errors, multibit processor storage errors, and storage protect key failures are handled as follows:

1. When a machine check interrupt indicates that an uncorrectable or unretryable CPU error associated with VM/370 has occurred, the system operator will be informed of the error and the CPU will be placed in a disabled wait state. If the error is associated with a virtual machine, the user is informed of the error and the virtual machine is terminated.
2. Damaged storage areas associated with a virtual machine page cannot be refreshed by VM/370 if the page has been altered since last brought into real storage. In that case, the user is informed of the failure and the virtual machine is reset but not logged off. If no alternation had occurred, refreshing is accomplished by marking the page as unavailable and allowing the normal paging facilities of VM/370 to refresh the page when the instruction is re-executed. Damaged storage areas associated with any portion of the VM/370 nucleus itself cannot be refreshed; multibit processor storage errors in VM/370 cause the real computing system to be placed in a disabled wait state.
3. When intermittent storage protect key failures occur, whether associated with VM/370 or a virtual machine, the key will be refreshed and operation will continue. If the storage protect key failure is uncorrectable and is associated with a virtual machine, the user is notified and the virtual machine is reset. Uncorrectable storage protect key failures associated with VM/370 cause the real computing system to be placed in a disabled wait state.

After storage recovery operations are completed by VM/370, a failing real storage block is removed from further use if program tests indicate that it should be. A subsequent loading of VM/370 restores the block for use.

The channel check handler receives control after a real channel error occurs. The error is recorded. Channel data checks associated with a virtual machine input/output operation are passed on to the virtual machine for handling; other types of channel errors cause termination of the virtual machine. Channel errors associated with an input/output event initiated by VM/370 (for example, paging and spooling operations) are retried by the appropriate device-dependent error recovery procedure if the error is considered recoverable. System termination will result if the channel has been reset, if the device address stored is invalid, or if a previous channel error cannot be successfully recorded.

Error Recovery Procedures

Device-dependent error recovery procedures are included in VM/370 for all devices supported by VM/370. Functionally, these procedures perform as do their counterparts in an OS or DOS system. The standards used by OS or DOS for priority of error testing, recommended retry action, and number of retry attempts for a particular error type are used by VM/370. The function of the error recovery procedures include the capability to accept and use the extended channel status word, determine if retry is possible, and initiate retry actions.

An appropriate error recovery procedure is invoked whenever an error occurs which is related to a VM/370 input/output event. If the error cannot be corrected, error recording occurs. To allow possible continued system operation of VM/370 in the event of a catastrophic error on one of the primary system devices, the computer operator may specify to VM/370 that further retry action should be attempted.

Input/output errors (excluding channel control checks and interface control checks) associated with virtual machine START I/O requests are passed to the virtual machine. VM/370 will record equipment checks for virtual machines. It is the responsibility of the virtual machine operating system to assess the error and attempt retry. Note that CMS uses the DIAGNOSE interface to request VM/370 to perform input/output operations, and VM/370 then performs any necessary recovery operations for errors associated with the request.

Recording Facilities

The CP Environment Recording, Edit, and Print program (CPEREP) is included as part of the VM/370 system and is run as a problem program under CMS. The printed output from CPEREP under VM/370 has the same format as that generated by OS. The CPEREP program (1) edits and prints all error records contained on the system error recording file, and (2) creates a history of records on an accumulation tape that is used as input to programs which run under OLTSEP as described in the following text.

VM/370 REPAIR FACILITIES

The Online Test Standalone Executive Program (OLTSEP) and Online Tests (OLT) execute in a virtual machine that runs concurrently with normal system operations. These programs provide online diagnosis of input/output errors for most devices that attach to the IBM System/370.

The IBM Customer Engineer can execute online tests from a terminal as a user of the system; VM/370 console functions, including the ability to display or alter the virtual machine storage, are available when these tests are run. Those tests that violate VM/370 restrictions may not run correctly in a virtual machine environment.

VM/370 RESTART FACILITIES

Whenever a system failure causes an abnormal termination of the real machine, not resulting in a disabled wait state, VM/370 automatically dumps and reloads itself. No operator intervention is required. The system will attempt to execute a warm start, thus allowing terminal users to be reconnected automatically and completed spool files to be maintained. In the event of a warm start, any device reconfiguration that had been performed by the real computing system operator will be remembered by VM/370; storage reconfiguration data, which is acquired during the process of recovering from real storage errors, will be lost. After a VM/370 system failure, each user must re-access VM/370; each virtual machine operating system must be reloaded.

The resetting of a virtual machine, whether caused by a real computing system malfunction or a virtual machine program error, does not affect the execution of any other virtual machine.

System Requirements

MACHINE REQUIREMENTS

The following machines and devices will be supported by VM/370. Specific information regarding model numbers and supported features can be provided by an IBM marketing representative.

CPU's

IBM System/370 Model 135
IBM System/370 Model 145
IBM System/370 Model 155 II
IBM System/370 Model 158
IBM System/370 Model 165 II
IBM System/370 Model 168

These machines must be equipped with the Dynamic Address Translation facility, the System Timing facilities, and at least 245,760 bytes of real main storage. The Models 135 and 145 require the floating-point feature. The Models 165 II and 168 require the Channel Indirect Data Addressing feature on each of the following standalone channels: 2860, 2870, and 2880.

The resident portion of the VM/370 control program requires approximately 80K bytes of real main storage plus an average of 2K bytes per active virtual machine.

System Consoles

The following system consoles as well as the terminals listed below are supported by VM/370 as real and virtual machine operators consoles.

- IBM 3210 Console Printer-Keyboard, Models 1 and 2
- IBM 3215 Console Printer-Keyboard, Model 1
- IBM 2150 Console with 1052 Printer-Keyboard, Model 7
- IBM 3155 System Console (in 3215 Emulator Mode) with the 3213 Printer Model 1 required
- IBM 7412 Console (via RPQ AA2846) with a 3215, Model 1

Note: The IBM 3066 System Console for the System/370 Models 165 II and 168 is not supported by VM/370 for its own use. It can, however, be attached to an OS virtual machine for use as an OS operator's console. VM/370 requires one of the following for use as the VM/370 system console:

- A local terminal (via control unit)
- A 1052, Model 7 (via the 2150 Console)
- A 3215, Model 1 (via the 7412 RPQ Console)

Only the latter two devices (the system consoles) can be used to generate the *initial* VM/370 system on a Model 165 II or 168; therefore, the initial VM/370 generation must be performed on a suitably equipped System/370.

Direct Access Storage Devices

IBM 2314 Direct Access Storage Facility
IBM 2319 Disk Storage
IBM 3330 Disk Storage, Model 1
IBM 3333 Disk Storage, Model 1
IBM 2305 Fixed Head Storage, Model 1 (Models 165 II and 168 only)
IBM 2305 Fixed Head Storage, Model 2

Direct Access Control Units

IBM 3345 Integrated Storage Control, Models 3, 4, 5 on the System/370 Model 145 for 3330
IBM 2835 Storage Control, Model 1 (Models 165 II and 168 only)
IBM 2835 Storage Control, Model 2
IBM 2844 Auxiliary Storage Control
IBM 3830 Storage Control, Models 1 and 2
IBM Integrated File Adapter (#4650) on the System/370 Models 135 and 145 for 2319
IBM Integrated File Adaptor on the System/370 Model 135 for 3330, Model 1 and 3333, Model 1
IBM Integrated Storage Control on the System/370 Model 155 II for 3330, Model 1 and 3333, Model 1
IBM Integrated Storage Control on the System/370 Model 165 II for 3330, Model 1 and 3333, Model 1

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

Magnetic Tapes

IBM 2401, 2402, 2403 Magnetic Tape Units
IBM 2415 Magnetic Tape Unit, Models 1, 2, 3, 4, 5, 6
IBM 2420 Magnetic Tape Unit, Models 5, 7
IBM 3420 Magnetic Tape Unit, Models 3, 5, 7

Magnetic Tape Control Units

IBM 2804 Tape Control
IBM 2803 Tape Control
IBM 3803 Tape Control

Printers

IBM 1403 Printers, Models 2, 3, 7, and N1
IBM 1443 Printer, Model N1
IBM 3211 Printer

Readers/Punches

IBM 2501 Card Reader
IBM 2540 Card Read Punch
IBM 3505 Card Reader
IBM 3525 Card Punch

Unit Record Control Units

IBM 2821 Control Unit
IBM 3811 Printer Control Unit

Telecommunications

Terminals

The following devices are supported as virtual machine operator consoles (and, consequently, as CMS user terminals):

IBM 2741 Communications Terminal
IBM 1050 Data Communication System
Line Control for CPT-TWX (Model 33/35) terminals

The following device is supported for remote spooling and the entry of spool commands:

IBM 2780 Data Transmission Terminal Model 2

Transmission Control Units

IBM 2701 Data Adapter Unit
IBM 2702 Transmission Control
IBM 2703 Transmission Control
IBM 3705 Communications Controller (In 2701, 2702, 2703 emulation mode) *Note:* The 3705 Emulator must be generated, loaded, and dumped by a DOS, DOS/VS, OS, or OS/VS virtual machine.
IBM Integrated Communications Adapter (#4640) available on the System/370 Model 135

Minimum VM/370 Configuration

CPU	One of the System/370 Models designated.
Storage	245,760 bytes
One	Console device
One	Printer
One	Card Reader
One	Card Punch
Two	Spindles Direct Access Storage
One	Nine-track Magnetic Tape Unit
One	Telecommunications Control Unit (or the Integrated Communications Adapter on the System/370 Model 135)
One	Multiplexer Channel
One	Selector or Block Multiplexer Channel
One	Communications Terminal

VM/370 PROGRAMMING CHARACTERISTICS

VM/370 has been written in VM/370 System Assembler language, using the instructions available only on an IBM System/370 with dynamic address translation. CMS command modules are written in VM/370 System Assembler language.

MAINTENANCE CONSIDERATIONS

All program releases and program fixes use CMS as an installing and update vehicle.

Appendix A. IBM Programs Executable Under CMS

Certain facilities mentioned in this publication must be separately ordered from IBM. The following table lists the language program products that are executable under the CMS component of VM/370.

Program Product	*Program Number*
OS Code & Go FORTRAN	5734-FO1
OS FORTRAN IV (G1)	5734-FO2
OS FORTRAN IV Library (Mod I)	5734-LM1
OS FORTRAN IV (H Extended)	5734-FO3
OS FORTRAN IV Library (Mod II)	5734-LM3
FORTRAN Interactive Debug	5734-FO5
OS Full American National Standard COBOL Version 4 Compiler and Library	5734-CB2
OS Full American National Standard COBOL Version 4 Library	5734-LM2
OS PL/I Optimizing Compiler	5734-PL1
OS PL/I Resident Library	5734-LM4
OS PL/I Transient Library	5734-LM5
OS PL/I Optimizing Compiler and Libraries	5734-PL3

The following processor is distributed with VM/370 as a component of the system:

VM/370 System Assembler

The IBM System/360 and System/370 operating systems discussed in the text are not components of VM/370 and must be ordered separately. These include DOS, OS, and PS44. The Conversational Monitor System (CMS) is a component of VM/370 and is distributed with it.

The following is distributed with VM/370 as a convenience to the customer but is not part of the VM/370 System Control Program:

- A BASIC language facility consisting of the CALL-OS BASIC Compiler and Execution Package, Version 1.1, adapted for use with CMS.

This BASIC facility will receive Class A maintenance by the VM/370 Central Programming Service.

Appendix B. System Commands

An overview of the command facilities available to VM/370 users is given in the following text. Not all system commands are included here. Two command categories exist, the control program console functions and the CMS commands. The console functions may be used at any time, without regard to which operating system is controlling the user's virtual machine. To issue these console functions, the user must first suspend execution in the virtual machine by pressing the attention key on his terminal twice; this is the virtual machine equivalent of pressing the stop button on a real computing system.

The commands of the Conversational Monitor System form the second category of commands. These provide conversational facilities for file manipulation, program compilation and execution, and general system control. The user's virtual machine must be running CMS if these commands are to be used. The VM/370 console functions are available to the CMS user without leaving the CMS environment, that is, the attention key need not be depressed.

VM/370 CONSOLE FUNCTIONS

The console functions of VM/370 are a set of interactive commands used to control the real computing system and VM/370, and to provide user control of virtual machines and associated control program facilities.

Each user of VM/370 is provided with one or more privilege classes as part of the directory entry of his virtual machine. A user's privilege class defines his allowable subset of VM/370 console functions. Seven privilege classes exist:

1. System operators having overall control of the real computing system.
2. Operators having resource control of the real computing system.
3. System programmers having limited functional control of VM/370 parameters.
4. Spooling operators having control of the unit record equipment and the spool files.
5. System analysis users permitted to inspect certain VM/370 statistical data.
6. Customer engineering users having responsibility for machine maintenance.
7. General users.

The last class permits all of the functions needed to control normal virtual machine operation and communication.

General Users

In order to become active in the system, the user prepares his terminal and establishes a line connection to the real computing system that is running VM/370. At that point, the LOGIN command identifies the user to VM/370 which then creates the control blocks necessary to simulate the virtual machine configured in the directory entry for the user. Alternately, the user's terminal may become a remote terminal for a virtual machine operating system which is multi-access (such as DOS-APL). To identify himself to that kind of system, the user issues the DIAL command and is, thereafter, controlled by the multi-access system directly; the user's terminal must be of a type supported by the multi-access system.

Ordinarily, the user will immediately load an operating system into his virtual machine. The IPL command is used to do this. Execution in the virtual machine may be stopped at any time, and the user may then invoke commands to simulate the functions of the operator's console or otherwise to control his virtual machine. These include:

Command	*Meaning*
EXTERNAL	Cause an external interrupt
ADSTOP	Define an instruction address stop location in virtual storage
DISPLAY	Type specified virtual machine registers or virtual storage contents in hexadecimal or EBCDIC
DETACH	Remove a specified device from the virtual machine configuration
READY	Simulate a device end interrupt from a virtual device
SPOOL	Alter the spooling control options (such as number of copies) for one or more virtual unit record devices that are controlled by the spooling facilities. In addition, this command is used to transfer data files among users.
STORE	Insert data into virtual machine registers or virtual storage
BEGIN	Resume execution in the virtual machine (the functional equivalent of pressing the Start key on a real computing system)
LINK	Make a specified virtual direct access storage device a part of the virtual machine configuration if the device is defined as

Command	*Meaning*
LINK (*cont.*)	sharable and the user can supply the appropriate password
QUERY	Interrogate certain system information such as the log message, the number of spool files, or the virtual machine configuration
SET	Establish certain system values such as the level of error message to be printed or the amount of VM/370 editing for terminal input lines

Other Users

System operator users may use the SET command to provide any of the VM/370 performance options, except virtual=real, to a particular virtual machine. (The virtual=real option is defined in the VM/370 system generation and specified in the directory entry of the selected system.) System operators control the orderly activity of the real computing system with FORCE to terminate a particular virtual machine, SHUTDOWN to terminate all VM/370 functions such that a warm start can be performed, and ACNT to create accounting records for all active users.

Operator users may modify the real communication line characteristics with ENABLE and DISABLE, logically remove a device from the real computing system with DETACH and VARY, and add a device to either the real computing system or a virtual machine configuration with ATTACH. SAVESYS may be issued to save on VM/370 direct access space a virtual machine's storage space, registers, and program status word.

Spooling operator commands include BACKSPAC, FLUSH, PURGE, REPEAT, SPACE, HOLD, and FREE. System analysis users may issue DCP to type real storage locations at the terminal or DMCP to print them offline. They may use QUERY to learn current system paging characteristics. System programmer users may define a paging characteristic in SET which will affect the amount of multiprogramming attempted by VM/370.

CMS COMMANDS

The following commands are available only to users of CMS. They, together with VM/370 commands, provide time-sharing services in areas of problem solving, program development, and general interactive work.

File Manipulation Commands

The creation and maintenance of user files is an important function of CMS. Files can be created on disk and so made accessible for other command processing by use of the commands READCARD, DISK, and TAPE. Files can be created directly from terminal keyboard input through use of the EDIT command, which invokes a powerful context and line editor. General file manipulation commands include COPYFILE, ERASE, and MOVEFILE. The FILEDEF command is used by the compiler interface programs of CMS to define the location and data characteristics of data sets for the OS simulation routines; it may be used by any terminal user to provide device independence since it simulates the function of the Data Definition card of OS Job Control Language.

A user makes a virtual disk active and defines its mode with the ACCESS command. The virtual disk must have been previously made part of the virtual machine configuration via the directory entry or the VM/370 LINK or ATTACH commands. A virtual CMS disk must have been initialized with the CMS FORMAT command that writes a home address and a device-dependent number of records containing zeros on each track.

Files may be passed to the spooling facilities of VM/370 and scheduled for later output by the PRINT and PUNCH commands. The TYPE command will print all or part of a file at the terminal. Status information about user files, for example, file size and date last written, can be written to the terminal or into a CMS file with the LISTFILE command.

Program Compilation and Execution Commands

The compilers executable under CMS are invoked by name and provided with a source file whose filetype designator is indicative of the compiler. The commands are ASSEMBLE, BASIC, COBOL, FORTGI, FORTHX, GOFORT, and PLIOPT. Each allows the specification of a parameter list that may contain CMS options as well as compiler options which are identical to those coded on an OS EXEC card when invoking the compiler from OS.

Program execution is controlled by CMS commands such as RUN, INCLUDE, and LOAD. A "core-image" copy of the program can be recorded on disk with GENMOD, and later retrieved for execution with LOADMOD. Libraries to be searched during program compilation or load are specified by the GLOBAL command.

Control Commands

The CMS user is able to define certain system parameters with SET. The parameters include the amount of information included in the message printed at the end of command processing, the type of error messages to be printed at the terminal, and whether unknown commands should be passed on to VM/370; with QUERY the user is given the current status of these and other CMS parameters. Command names may be altered by a user via entries in a named file with a SYNONYM filetype. The EXEC command specifies a file of CMS commands, as well as logic and control statements; which will be executed in a predetermined sequence by the EXEC processor of CMS.

Other Commands

Debugging facilities of CMS are invoked by the DEBUG and SVCTRACE commands. These facilities are in addition to those provided by VM/370 in its TRACE command. Two service programs of VM/370 execute in CMS but provide service to all system users. These are MINIDASD which initializes minidisks for OS or DOS use, and DDR which provides general dump/restore services for minidisks of any type. (The CMS FORMAT command initializes minidisks for CMS.)

Appendix C. VM/370 Restrictions

A virtual machine created by VM/370 is capable of running an IBM System/360 or System/370 operating system as long as certain VM/370 restrictions are not violated. Virtual machine restrictions and certain execution characteristics are stated in this appendix.

In general, virtual machines may not execute channel programs that are dynamically modified, that is, channel programs changed between the time the START I/O (SIO) is issued and the end of the input/output operation occurs; or those changed by the channel program itself or by the CPU. However, the OS Indexed Sequential Access Method (ISAM), which uses a self-modifying channel program for some of its operations, receives special handling if VM/370 is generated with the ISAM option and the virtual machine using ISAM has that option selected in the directory description. The restriction against dynamically modified channel programs does not apply if the virtual machine has the virtual=real performance option.

In addition to the above restriction of dynamically modified input/output sequences, there are the following restrictions for minidisks:

1. In the case of READ HOME ADDRESS with the skip bit off, VM/370 will modify the home address data in user storage at the completion of the channel program because the addresses must be converted for minidisks; therefore, the data buffer area may not be dynamically modified during the input/output operation.
2. On a minidisk, if a CCW string uses multitrack search on input/output operations, subsequent operations to that disk must have preceding seeks or continue to use multitrack operations. There is no restriction for dedicated disks.
3. OS ISAM may be used with a minidisk only if the minidisk is located at the beginning of the physical disk (that is, at cylinder zero).
4. VM/370 will not return an end of cylinder condition to a virtual machine which has a virtual 2311 mapped to the top half (that is, tracks 0 through 9) of 2314 or 2319 cylinders.

Timing dependencies in input/output devices or programming will not function consistently under VM/370:

1. Programming that makes use of the PCI channel interrupt for channel program modification or processor signalling must be written so that processing can continue normally if the PCI is not recognized until input/output completion or if the modifications performed are not executed by the channel.
2. Devices that expect a response to an interrupt within a fixed period of time may not function correctly because of execution delays caused by normal VM/370 system processing. An example of such a device is the IBM 1419 Magnetic Character Reader.
3. The operation of a virtual block multiplexer channel is timing dependent. For this reason, the channel will appear available to the virtual machine operating system, and channel available interrupts will not be observed. However, operations on virtual block-multiplexing devices should use the available features like rotation position sensing to enhance utilization of the real channels. If a virtual machine has a dedicated block multiplexer channel, then that channel will operate in true block-multiplexing mode for the virtual machine.

Programs written for CPU model-dependent functions may not execute properly in the virtual machine under VM/370:

1. Programs written to examine the machine logout area will not have meaningful data since VM/370 will not reflect the machine check logout data to a virtual machine.
2. Programs written to obtain CPU or channel identifications will receive the real machine value.
3. No simulation of other CPU models is attempted by VM/370.

Other characteristics that exist for a virtual machine under VM/370 are as follows:

1. If the virtual=real option is selected for a virtual machine, input/output operations specifying data transfer into or out of the virtual machine's page zero, or into or out of storage locations whose addresses are greater than the storage allocated by the virtual=real option, must not occur. The storage-protect-key mechanism of the IBM System/370 CPU and channels is operative in these situations but unable to provide predictable protection to other virtual machines. In addition, violation of this restriction may compromise the integrity of the system. The results are unpredictable.
2. VM/370 has no multiple path support and, hence, will not take advantage of the two channel switch. However, a two channel switch can be used between

the IBM System/370 running a virtual machine under VM/370 and another CPU.

3. The DIAGNOSE instruction cannot be issued by the virtual machine for its normal function. VM/370 uses this instruction to allow the virtual machine to communicate system services requests.
4. The following telecommunication access methods (or the designated option) violate the restriction on timing dependency by using program controlled interrupt techniques and/or the restriction on dynamically modified channel programs:

   - OS Basic Telecommunications Access Method (BTAM) with the dynamic buffering option
   - OS Queued Telecommunications Access Method (QTAM)
   - DOS Queued Telecommunications Access Method (QTAM)
   - OS Telecommunications Access Method (TCAM)

   These access methods require the virtual machine to have the virtual=real and other performance options. (OS BTAM can be generated without dynamic buffering, in which case no virtual machine execution violations occur.)
5. A control unit never appears busy to a virtual machine unless the unit is on a channel dedicated to the virtual machine.
6. The number of pages into which input/output is being performed must not exceed the total number of pageable page frames available in real storage; violation of this restriction will cause the real computing system to be put into an enabled wait state.
7. The VM/370 console function IPL cannot simulate self-modifying IPL sequences off dedicated unit record devices or certain self-modifying IPL sequences off tape devices.
8. The interval timer provided by VM/370 does not perform in the same manner as the standard interval timer. The real timer that is given to a virtual machine by specifying the real timer option in the directory reflects the execution time plus the wait time for that virtual machine. If a virtual machine does not have the real time option, its timer reflects only the execution time used.
9. The VM/370 spooling facilities do not support the following: punch-feed-read, stacker selection and column binary operations. Detection of carriage control channels is supported for a virtual 3211 only.
10. VM/370 does not support count control on the virtual 1052 operator's console.
11. Programs that use the integrated emulators will function only if the real computing system has the appropriate compatibility feature. No simulation by VM/370 is attempted. The DOS emulator is not supported under VM/370.
12. A virtual machine in basic control mode is limited to seven I/O channels. The extended channel feature is not supported. A virtual machine in extended control mode is limited to 16 I/O channels.
13. The READ DIRECT and WRITE DIRECT instructions are not supported for a virtual machine.
14. The System/370 instruction of SET CLOCK cannot be simulated and, hence, is ignored if issued by a virtual machine. The System/370 instruction STORE CLOCK is a nonprivileged instruction and cannot be trapped by VM/370; it will provide the true TOD clock value from the real CPU.
15. Paper tape input/output is not supported for the virtual machine console; however, subject to the other restrictions, it will function if supported by the virtual machine operating system.
16. Virtual machines will not be able to utilize the indexing facility of the 3811 to electronically offset or displace 3211 print lines.

The following restrictions apply to CMS, the conversational subsystem of VM/370:

1. The maximum size of a CMS minidisk on a 3330 disk is 246 cylinders.
2. Unit record equipment cannot be dedicated to CMS; the spooling facilities of VM/370 must be used.
3. Executing OS programs produced by language processors under CMS is restricted to using only those OS facilities that have been simulated in CMS.
4. CMS will not execute standalone on a real IBM System/370 CPU.

Appendix D. VM/370 Compatibility

VM/370 and its associated component, the Conversational Monitor System, is based on the CP-67/CMS system and is designed especially for the IBM System/370. Dynamic address translation provides the same facilities as did dynamic address translation (the DAT box) on the System/360 Model 67, but differs in design detail and implementation. Consequently, CP-67/CMS will not run on a System/370 and VM/370 will not run on a System/360. The internal structure of VM/370 and the control blocks used differ from the structure and control blocks of CP-67/CMS. User modifications to CP-67/CMS should be re-evaluated considering the new facilities of VM/370 and, if still desirable, redesigned for VM/370.

The Conversational Monitor System of VM/370 functionally extends the Cambridge Monitor System of CP-67/CMS. The following should be fully understood by customers planning conversion to VM/370:

1. The CMS disk file formats are unchanged. CMS provides a utility command to copy files from 2314 or 2319 to 3330, or from the 2311 to the 2314 or 3330. User files, which currently exist on 2311, cannot be accessed directly by the Conversational Monitor System, and should, therefore, be dumped to tape and then reloaded or copied onto a 2314 or 2319 using the Cambridge Monitor System.
2. The Conversational Monitor System supports ten active disks, with mode letters A-G, S, Y, and Z. The order of search used is alphabetic by mode letter. Existing CMS procedures that depend upon another order of search must be changed.
3. If a CMS command has been issued from a program and that command's format has changed, then the PLIST entries in the program must be modified.
4. The Conversational Monitor System utilizes the VM/370 support for shared segments. For this and other reasons, the CMS nucleus extends beyond location 12000 (hexadecimal). Consequently, all existing MODULE files must be recreated for VM/370.
5. Filename and filetype naming conventions have been changed. Existing file identifications may need to be altered prior to use under the CMS component of VM/370.
6. TXTLIB files must be regenerated because the dictionary format has changed.

A more comprehensive list of data processing terms may be found in the publication *IBM Data Processing Glossary,* GC20-1699.

active page: A page in real storage that can be addressed.

address translation: The process of changing the address of a data item or an instruction from its virtual address to its real storage address. *See also* dynamic address translation.

attention key interrupt handling: Effects a transfer of control between VM/370 and a virtual machine operating system.

basic control mode: A mode in which the features of an IBM System/360 computing system and additional System/370 features, such as new machine instructions, are operational on a System/370 computing system. *See also* extended control (EC) mode.

BC mode: See basic control mode.

channel program translation: Replacement by VM/370 of virtual addresses in a channel program with real addresses.

command privilege class: One of seven logical subsets of VM/370 commands or console functions.

console function: A VM/370 facility whereby the remote terminal user can simulate a function available at a System/370 console. The command facilities of VM/370 (excluding the subsystem CMS) are referred to collectively as console functions.

context editing: A method of editing a line data set or file without using line numbers. To refer to a particular line, all or part of the contents of that line are specified.

DAT: See dynamic address translation.

dedicated channel option: A virtual machine option which improves performance in a virtual machine by bypassing address translation of virtual devices and VM/370 channel scheduling.

demand paging: Transfer of a page from external page storage to real storage at the time it is needed for execution.

directory file: A VM/370 disk file which defines the virtual machine configuration of each user.

dynamic address translation (DAT): The change of a virtual storage address to a real address during execution of instruction. *See also* address translation.

extended control (EC) mode: A mode in which all the features of an IBM System/370 computing system, including dynamic address translation, are operational. *See also* basic control (BC) mode.

favored execution option: A virtual machine performance option which attempts to provide a specific percentage of real CPU time to a particular virtual machine.

line number editing: A mode of operation under the EDIT command in which lines or records to be modified are referred to by line or record number.

minidisk: A contiguous portion of direct access storage which is assigned a virtual device address.

page: (1) A fixed length block of instructions, data, or both, that can be transferred between real storage and external page storage. (2) To transfer instructions, data, or both, between real storage and external page storage.

page frame: A block of real storage that can contain a page.

page table: A table which indicates whether a page is in real storage and correlates virtual with real storage addresses.

paging: The process of transferring pages between real storage and external page storage.

paging device: A direct access device on which pages (and possibly other data) are stored.

preferred virtual machine: A generic name describing a particular virtual machine to which one or more of the VM/370 options have been assigned.

priority: A virtual machine parameter which influences the internal scheduling algorithm of the VM/370 control program.

real address: The address of a location in real storage.

reserved page frame option: A virtual machine option which allows the most active pages of a virtual machine's storage to remain allocated in real storage.

segment table: A table used in dynamic address translation to control access to virtual storage segments. Each entry indicates the length, location, and availability of a corresponding page table.

shadow page table: A page table created and used by VM/370 to control the execution of a virtual machine operating system which is performing paging operations.

spooling area: Any direct access storage area temporarily used by VM/370 to store input for a virtual card reader or output for a virtual printer or punch.

virtual address: An address that refers to virtual storage and must, therefore, be translated to a real storage address when it is used.

virtual computing system: A synonym for virtual machine.

virtual machine: The functional equivalent of an IBM System/370 computing system. Each virtual machine is controlled by a suitable operating system. VM/370 controls the concurrent execution of multiple virtual machines.

virtual=real option: A virtual machine option which permits the entire storage of a virtual machine to have the same range of addresses as real storage, excepting page zero which is relocated. It is used for programs which dynamically modify channel programs and for performance gains.

virtual storage: (1) Addressable space that appears to the user as real storage, from which instructions and data are mapped into real storage locations. The size of virtual storage is limited by the addressing scheme of the computing system and by the amount of auxiliary storage available, rather than by the actual number of real storage locations. (2) Addressable space that appears to the user as real storage, from which instructions and data are mapped into real storage locations. The size of the virtual storage is limited by the addressing scheme of the virtual machine and the aggregate amount of external page storage available.

warm start: The automatic reinitialization of the VM/370 control program that occurs if the control program cannot continue processing.

INDEX

access
    multi 15, 16
address translation, dynamic 5, 6, 11, 15, 24, 35
addressing, channel indirect data 12, 24
attention key 29

basic control mode 7, 34, 37
BASIC program 27

Cambridge Monitor System 17, 35
central processing unit *see* CPU, virtual
channel
    check handler 21, 22
    dedicated 13, 16
    handler, check 21, 22
    indirect data addressing 12
    virtual 5, 7
channel check handler 21, 22
channel, dedicated 13, 16
channel indirect data addressing 12, 24
channel, virtual 5, 7
check
    channel, handler 21, 22
    handler
        channel 21; 22
        machine 21
    machine, handler 21
check handler
    channel 21, 22
    machine 21
class
    command, privilege 10, 14, 29
    privilege 10, 14, 31
    privilege, command 10, 14, 29
class, privilege 10, 14, 31
class, command privilege 10, 14, 29
CMS
    CP-67/CMS 5, 17, 35
    file, user 18-19, 30
    flip-flop 9, 19-20
    user file 18-19, 32
CMS 9, 19-20
    see also Conversational Monitor System
command
    class, privilege 10, 14, 29
    privilege class 10, 14, 29
commands
    CP *see* console functions
computing system virtual 5
concurrent execution 5

configuration, virtual machine 5, 6, 7, 24, 25
console
    functions (CP commands) 6, 13, 14, 17, 29-30
        accessing from CMS 17
        general users 29, 30
        other users 30
        privilege classes of 10, 14, 29
        summary of 29, 30
    operator's, virtual 5, 6, 15
    virtual operator's 5, 6, 15
console functions 6, 13, 14, 17, 29-30
console, virtual operator's 5, 6, 15
context editor 18, 19, 30
control
    mode
        basic 7, 34, 37
        extended 5, 7, 34, 37
    program (CP) 5, 10
control mode
    basic 7, 34, 37
    extended 5, 7, 34, 37
control program (CP) 5, 10
CMS flip-flop 9, 19-20
CMS user file 18-19, 30
conversational 5, 8, 15, 17, 19
Conversational Monitor System 5, 6, 7, 8, 13, 17-20, 30-31, 34
CP commands 6, 13, 14, 17, 29-30
CP-67/CMS 5, 17, 35
CPU, virtual 5, 7

data
    addressing, channel indirect 12, 24
    channel indirect, addressing 12, 24
debugging 8, 17, 20, 31
dedicated
    channel 13, 16
    device 12, 13
development, program 8-9, 15, 17, 19-20
device, dedicated 12, 13
device, paging 11, 15
device, spooled 12
device(s)
    dedicated 12, 13
    spooled 12
    virtual 5, 7-8, 9, 12, 17
DIAGNOSE interface 7, 13, 22, 33-34
directory, user
    entry 6, 10, 11, 13, 17, 30
    file 5, 10

directory (entry) 6, 10, 11, 13, 17, 30
directory (file) 5, 10
Disk Operating System (DOS) 5, 6, 7, 9, 15
disk(s)
 sharing, virtual 11, 19
 virtual 7, 32
 virtual, sharing 11, 19
disk sharing, virtual 11, 19
disks, virtual 7, 32
dispatching 10
DOS (Disk Operating System), 5, 6, 7, 9, 15
dynamic address translation 5, 6, 11, 15, 24, 35

editor, context 18, 19, 30
emulator 34
entry, directory 6, 10, 11, 13, 17, 30
EXEC procedures 17, 31
execution
 concurrent 5
 favored 10, 16
extended control mode 5, 7, 34, 37

favored execution 10, 16
file
 CMS user 18-19, 30
 directory 5, 10
 spool 13
 transfer 13
 user, CMS 18-19, 30
filename 18
file transfer 13
flip-flop 9, 19-20
 CMS 9, 19-20
flow, work 5
frame(s)
 page 11, 12
 reserved page 11, 16
functions
 console 6, 13, 14, 17, 29-30
  accessing from CMS 17
  general users 29, 30
  other users 32
  privilege classes of 10, 14, 29
 CP commands *see* console functions

handler
 channel check 21, 22
 machine check 21

identification, user 6, 10
indirect data addressing, channel 12
input/output operations 8, 11-13
interface, DIAGNOSE 7, 13, 22, 33-34
interrupts 10
interrupts, reflection of 10, 13

key, attention 29

languages 17, 27
login 10, 11

machine
 check handler 21
 configuration, virtual 5, 6, 7, 24, 25, 29, 30
 handler, check 21
 operating system, virtual 7, 12, 15-16, 23
 virtual 5, 6, 9, 24, 25, 29-30, 33-34
  configuration 5, 6, 7, 24, 25, 29, 30
  operating system 7, 12, 15-16, 23
machine check handler 21
macro simulation, OS 19
management, virtual storage 6-7, 10-11
MFT *see* OS/MFT
minidisk 7, 9, 12, 18, 31, 33
 restrictions 31, 33
mode
 basic control 7, 34, 37
 control
  basic 7, 34, 37
  extended 5, 7, 34, 37
 extended control 5, 7, 34, 37
Monitor
 Cambridge, System 17, 35
 Conversational, System 5, 6, 7, 8, 13, 17-20
 30-31, 34
 System
  Cambridge 17, 35
  Conversational 5, 6, 7, 8, 13, 17-20, 30-31, 34
multi
 access 15, 16
 programming 5
 task 7, 15
multi-access 15, 16
multiprogramming 5
multitask 7, 15
MVT *see* OS/MVT

operating
 system(s) 5, 7, 8, 15-16, 29
 system, virtual machine 7, 12, 15-16, 23
Operating System *see* OS
operating system(s) 5, 7, 8, 15-16, 29
operations
 input/output 8, 11-13
 input 8, 11-13
 output 8, 11-13
operator's console, virtual 5, 6, 15
OS 8, 9, 16, 30, 33, 34
 macro simulation 19
 OS/MFT 7
 OS/MVT 7, 15
 OS/VS1 7, 10, 15
OS/MFT 7

OS/MVT 7, 15
OS/VS1 7, 10, 15
OS macro simulation 19
output *see* input/output

page 10, 11
  frame 11, 12
  frames, reserved 11, 16
  reserved, frames 11, 16
  table, shadow 11
  tables 10, 11
page frame 11, 12
page tables 10, 11
paging 11, 13, 16, 22
  device 11, 15
paging device 11, 15
password 6, 11
performance 11, 14, 15-16
priority 10
privilege
  command, class 10, 14, 29
  class, command 10, 14, 29
  class 10, 14, 29
privilege class 10, 14, 29
problem-solving 17
procedures, EXEC 17, 31
program
  control (CP) 5, 10
  development 8-9, 15, 17, 19-20
  products 27
  test 9
program development 8-9, 15, 17, 19-20
program test 9

real storage 6, 7, 11, 15, 22, 34
reflection of interrupts 10, 13
reserved page frames 11, 16

saved system 16
segment
  sharing 16
  tables 10, 11
segment sharing 16
segment tables 10, 11
shadow page table 11
sharing
  segment 16
  time 6, 8, 17
  virtual disk 11, 19
sharing, virtual disk 11, 19
simulation, OS macro 19
single
  task 7
  user system 8
  system user 8
single task 7
single user system 8
slicing, time 7, 10, 15
solving
  problem- 17
spool file 13
spooled device 12
spooling 9, 13, 17, 19, 29, 30
start, warm 23
storage
  real 6, 7, 11, 15, 22, 34
  management, virtual 6-7, 10-11
  virtual 5, 6-7, 8, 9, 10-11, 17, 29
  virtual, management 6-7, 10-11
System
  Cambridge Monitor 17, 35
  Conversational Monitor 5, 6, 7, 8, 13, 17-20, 30-31, 34
  Disk Operating (DOS) 5, 6, 7, 9, 15
  Monitor
    Cambridge 17, 35
    Conversational 5, 6, 7, 8, 13, 17-20, 30-31, 34
  Operating (OS) 8, 9, 16, 30, 33, 34
system(s)
  computing, virtual 5
  machine, operating, virtual 7, 12, 15-16, 23
  saved 16
  single user 8
  virtual, computing 5
  virtual machine operating 7, 12, 15-16, 23
  user, single 8

tables
  page 10, 11
  page, shadow 11
  segment 10, 11
  shadow page 11
task-single 7
terminal 5, 6, 8, 24, 25, 29
test, program 9
time
  sharing 6, 8, 17
  slicing 7, 10, 15
time-sharing 6, 8, 17
time slicing 7, 10, 15
transfer, file 13
translation, dynamic address 5, 6, 11, 15, 24, 35

user
  CMS 18-29, 30
  file, CMS 18-19, 30
  identification 6, 10
  single, system 8
  system, single 8
user identification 6, 10

virtual
  channel 5, 7

virtual (*cont.*)
computing system 5
configuration, machine 5, 6, 7, 24, 25, 29, 30
console, operator's 5, 6, 15
CPU 5, 7
devices 5, 7-8, 9, 12, 17
disks 7, 30
disk sharing 11, 19
machine 5, 6, 9, 24, 25, 29-30, 33-34
machine configuration 5, 6, 7, 29, 30
machine operating system 7, 12
management, storage 6-7, 10-11
operating system, machine 7, 12, 15-16, 23
operator's console 5, 6, 15
sharing, disk 11, 19
storage 5, 6-7, 8, 9, 10-11, 17, 24, 29
storage management 6-7, 10-11
system, computing 5
system, machine operating 7, 12, 15-16

virtual channel 5, 7
virtual computing system 5
virtual CPU 5, 7
virtual devices 5, 7-8, 9, 12, 17
virtual disks 7, 33, 34
virtual disk sharing 11, 19
virtual machine 5, 6, 9, 24, 25, 29-30, 33-34
virtual machine configuration 5, 6, 7, 24, 25, 33, 34
virtual machine operating system 7, 12, 15-16, 23
virtual operator's console 5, 6, 15
virtual storage 5, 6-7, 8, 9, 10-11, 17, 29
virtual storage management 6-7, 10-11
virtual=real 11, 15, 16, 30, 33
VS1 *see* OS/VS1

warm start 23
work flow 5

Trim Along This Line

READER'S COMMENTS

Title: IBM Virtual Machine Facility/370: Introduction

Order No. GC20-1800-0

Please check or fill in the items; adding explanations/comments in the space provided.

Which of the following terms best describes your job?

☐ Programmer	☐ Systems Analyst	☐ Customer Engineer
☐ Manager	☐ Engineer	☐ Systems Engineer
☐ Operator	☐ Mathematician	☐ Sales Representative
☐ Instructor	☐ Student/Trainee	☐ Other (explain below)

Does your installation subscribe to the SL/SS? ☐ Yes ☐ No

How did you use this publication?

☐ As an introduction	☐ As a text (student)
☐ As a reference manual	☐ As a text (instructor)

☐ For another purpose (explain) ____________________

Did you find the material easy to read and understand? ☐ Yes ☐ No (explain below)

Did you find the material organized for convenient use? ☐ Yes ☐ No (explain below)

Specific criticisms (explain below)

Clarifications on pages ____________________

Additions on pages ____________________

Deletions on pages ____________________

Errors on pages ____________________

Explanations and other comments:

GC20-1800-0

YOUR COMMENTS PLEASE . . .

This manual is one of a series which serves as a reference source for systems analysts, programmers, and operators of IBM systems. Your comments on the back of this form will be carefully reviewed by the persons responsible for writing and publishing this material. All comments and suggestions become the property of IBM.

Please note: Requests for copies of publications and for assistance in utilizing your IBM system should be directed to your IBM representative or to the IBM sales office serving your locality.

Trim Along This Line

FOLD FOLD

FIRST CLASS
PERMIT NO. 44308
CAMBRIDGE, MASS.

BUSINESS REPLY MAIL
NO POSTAGE STAMP NECESSARY IF MAILED IN U.S.A.

POSTAGE WILL BE PAID BY

IBM CORPORATION
VM/370 PUBLICATIONS
545 TECHNOLOGY SQUARE
CAMBRIDGE, MASS. 02139

FOLD FOLD

IBM Virtual Machine Facility/370: Introduction Printed in U.S.A. GC20-1800-0

IBM

International Business Machines Corporation
Data Processing Division
1133 Westchester Avenue, White Plains, New York 10604
[U.S.A. only]

IBM World Trade Corporation
821 United Nations Plaza, New York, New York 10017
[International]

GC20-1800-0

IBM Virtual Machine Facility/370: Introduction   Printed in U.S.A.   GC20-1800-0

IBM

International Business Machines Corporation
Data Processing Division
1133 Westchester Avenue, White Plains, New York 10604
(U.S.A. only)

IBM World Trade Corporation
821 United Nations Plaza, New York, New York 10017
(International)