File No. S370-36
Order No. SY20-0886-1

IBM Virtual Machine Facility/370: System Logic and Problem Determination Guide Volume 1

Systems

Control Program (CP)

Release 6 PLC 1

This publication is intended for the IBM system hardware and software support personnel. It provides the following information for the CP component of VM/370:

- Description of program logic
- Module descriptions and cross-references
- Abend and wait state codes

PREREQUISITE PUBLICATIONS

IBM Virtual Machine Facility/370:
Introduction, Order No. GC20-1800
Operator's Guide, Order No. GC20-1806
System Programmer's Guide, Order No. GC20-1807
Terminal User's Guide, Order No. GC20-1810
CP Command Reference for General Users, Order No. GC20-1820
IBM System/360 Principles of Operation, Order No. GA22-6821
IBM System/370 Principles of Operation, Order No. GA22-7000
IBM OS/VS, DOS/VS, and VM/370 Assembler Language, Order No. GC33-4010

| Second Edition (March 1979)

| This is a major revision of and obsoletes SY20-0886-0 and Technical
| Newsletters SN25-0446 and SN25-0467. This edition, SY20-0886-1,
| corresponds to Release 6 PLC 1 (Program Level Change) of the IBM Virtual
Machine Facility/370 and to all subsequent releases until otherwise
indicated in new editions or Technical Newsletters.

Technical changes and additions to text and illustrations are indicated by a vertical bar to the left of the change.

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

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

A form for readers' comments is provided at the back of this publication. If the form has been removed, comments may be addressed to IBM Corporation, VM/370 Publications, Dept. D58, Bldg. 706-2, P.O. Box 390, Poughkeepsie, New York 12602. IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation whatever. You may, of course, continue to use the information you supply.

Preface

This publication provides the IBM system hardware and software support personnel with the information needed to analyze problems that may occur on the IBM Virtual Machine Facility/370 (VM/370).

HOW THIS MANUAL IS ORGANIZED

This manual comprises three volumes:

"Volume 1. VM/370 Control Program (CP)," "Volume 2. Conversational Monitor System (CMS)," and "Volume 3. Remote Spooling Communications Subsystem (RSCS)" contain the logic description for each of the components. Each of these volumes is divided into four sections: Introduction, Method of Operation, Directory, and Diagnostic Aids.

The method of operation and program organization sections contain the functions and relationships of the program routines in VM/370. They indicate the program operation and organization in a general way to serve as a guide in understanding VM/370. They are not meant to be a detailed analysis of VM/370 programming and cannot be used as such.

The directories contain descriptions of all the assemble modules in CP, CMS, and RSCS. They also contain extensive cross-references between modules and labels within a VM/370 component.

The diagnostic aids sections contain additional information useful for determining the cause of a problem.

Appendix A, located in Volume 1, contains a description of VM/370 Extended Control-Program Support (ECPS).

Appendix B, also located in Volume 1, describes VM/370 support for the IBM 3850 Mass Storage System (MSS).

HOW TO USE THIS MANUAL

- Isolate the component of VM/370 in which the problem occurred.
- Use the list of restrictions in VM/370 System Messages to be certain that the operation that was being performed was valid.
- Use the directories and use the VM/370 Data Areas and Control Block Logic to help you to isolate the problem.
- Use the method of operation and program organization sections, if necessary, to understand the operation that was being performed.

DEVICE TERMINOLOGY

The following terms in this publication refer to the indicated support devices:

- "2305" refers to IBM 2305 Fixed Head Storage, Models 1 and 2.
- "270x" refers to IBM 2701, 2702, and 2703 Transmission Control Units or the Integrated Communications Adapter (ICA) on the System/370 Model 135.
- "3330" refers to the IBM 3330 Disk Storage, Models 1, 2, or 11; the IBM 3333 Disk Storage and Control, Models 1 or 11; and the 3350 Direct Access Storage operating in 3330/3333 Model 1 or 3330/3333 Model 11 compatibility mode.
- "3340" refers to the IBM 3340 Disk Storage, Models A2, B1, and B2, and the 3344 Direct Access Storage Model B2.
- "3350" refers to the IBM 3350 Direct Access Storage Models A2 and B2 in native mode.
- "3704", "3705", or "370X" refers to IBM 3704 and 3705 Communications Controllers.
- The term "3705" refers to the 3705 I and the 3705 II unless otherwise noted.
- "2741" refers to the IBM 2741 and the 3767, unless otherwise specified.
- "3270" refers to a series of display devices, namely the IBM 3275, 3276, 3277, 3278 Display Stations. A specific device type is used only when a distinction is required between device types.

Information about display terminal usage also applies to the IBM 3036, 3138, 3148, and 3158 Display Consoles when used in display mode, unless otherwise noted.

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

CP COMPONENT

PREREQUISITE PUBLICATIONS

IBM Virtual Machine Facility/370

Introduction, Order No. GC20-1800

Operator's Guide, Order No. GC20-1806

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

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

CP Command Reference for General Users, Order No. GC20-1820

COREQUISITE PUBLICATIONS

IBM Virtual Machine Facility/370

Planning and System Generation Guide, Order No. GC20-1801

System Messages, Order No. GC20-1808

OLTSEP and Error Recording Guide, Order No. GC20-1809

Operating Systems in a Virtual Machine, Order No. GC20-1821

Service Routines Program Logic, Order No. SY20-0882

Data Areas and Control Block Logic, Order No. SY20-0884

In addition, for EREP processing the following OS/VS Library publications are required:

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

OS/VS Environmental Recording Editing and Printing (EREP) Program Logic, Order No. SY28-0773

If the IBM 3850 Mass Storage System is attached, the following publications are required:

OS/VS Message Library: Mass Storage System (MSS) Messages, Order No. GC38-1000

IBM 3850 Mass Storage System (MSS) Principles of Operation: Theory, Order No. GA32-0035

IBM 3850 Mass Storage System (MSS) Principles of Operation: Reference, Order No. GA32-0036

SUPPLEMENTARY PUBLICATIONS

IBM System/360 Principles of Operation, Order No. GA22-6821

IBM System/370 Principles of Operation, Order No. GA22-7000

IBM 3270 Information Display System Components Description, Order No. GA27-2749

General Information Binary Synchronous Communications, Order No. GA27-3004

Contents

3203 MODEL 5 PRINTER SUPPORT

Changed: Documentation

VM/370 supports the 3203 Model 5 printer in the same manner as the 3203 Model 4 printer.

SHARED SEGMENT MODIFICATIONS

New: Program and Documentation

VM/370 now places the user in console function mode if he modifies a protected shared segment; it returns the modified page to free storage. VM/370 continues to give other users of the segment access to a fresh copy of the modified page. This is discussed in the "CP" Introduction" section of this publication.

The LOADSYS diagnose function is changed. When LOADSYS is executed, CP finds the system name table entry for the named system. In AP mode, two sets of page and swap tables are built, one for each processor. This is done for each shared segment in attached processor mode unless the named segment was defined as unprotected.

New abend codes are also added. These are found in the "CP Diagnostic Aids" section of this publication.

MESSAGE NO HEADER (MSGNOH) COMMAND SUPPORT

New: Program and Documentation

This support allows service virtual machines with privilege class B in a VM/370 system to send unformatted messages to other user of the system. The "CP Directories" section of this publication reflects the MSGNOH update.

PASSWORD ON-THE-COMMAND-LINE SUPPRESSION

New: Program and Documentation

This support is added to VM/370 to prevent disclosure of virtual machine and minidisk passwords on the same line as the LOGON, AUTOLOG, and LINK commands. To reflect this support, the DIAGNOSE Code X'08' instruction has been updated.

Changes are contained in the "CP Introduction" section of this publication.

DIAGNOSE CODE X'08' INSTRUCTION ENHANCED

Changed: Documentation

DIAGNOSE CODE X'08' has been modified to allow the caller to choose to have information returned to his virtual machine's buffer or handled as console line output. This information is found in the "CP Introduction" section of this publication.

LOGON, AUTOLOG AND LINK JOURNALING

New: Program and Documentation

VM/370 supports the journaling of LOGON and AUTOLOG commands that specify invalid passwords, and the journaling of all LINK commands. This is done via the generation of type 04, 05, and 06 accounting records.

The "CP Directories" section of this publication has been updated to reflect this information.

DIRECTORY UPDATE IN-PLACE

New: Program and Documentation

The DIAGNOSE Code X'84' instruction is added to VM/370 to allow certain directory options to be changed on-line without updating the directory source file.

This information is included in the "CP Introduction" and "CP Diagnostic Aids" sections of this publication.

VM/370 MEASUREMENT FACILITY (MONITOR) ENHANCEMENTS

New: Program and Documentation

The VM/370 Measurement Facility (Monitor) has been enhanced to extend its data collection and recording capabilities. These enhancements include:

- The gathering of additional data related to utilization of channels, devices, storage, alternate I/O paths, and AP/UP PROCESSING.
- Selective seeks that allow system programmers to select devices for which seek information is to be collected.
- Monitor-To-Disk support for real time allows system programmers to specify when the VM/370 Monitor is to close the spool file.

A new subroutine, DMKENTTI, is added. It is a high-frequency I/O status sampler that tests for busy conditions in all control units and devices by examining appropriate CP control blocks and in all channels via TCH instructions.

A new subroutine, DMKENT62 samples the data accumulated by DMKENTTI and writes it in a new class 6 code 2 record, after the standard class 6 (DASTAP) code 1 record has been collected in DMKMONTI.

This information is contained in the "CP Directories" section of this publication.

4331 AND 4341 PROCESSOR SUPPORT

New: Documentation

VM/370 provides support for the 4331 and 4341 processors and the 3278 Model 2A display station as a system console for these processors. The character set on the 3278 Model 2A is the same as on the current 3270 display systems, plus six additional graphic national usage characters that are acceptable in input and output data streams.

New modules and their entry points have been added to the "Label-To-Module" and "Module-To-Label" cross references in this publication.

SPECIAL MESSAGE FACILITY

New: Program and Documentation

A new CP command, SMSG, is provided to allow a terminal user to send a message to another user's virtual storage. The receiving virtual machine must be prepared to receive the message. This is done by issuing a VMCF AUTHORIZE and setting SMSG ON prior to receiving a message.

Information about Special Messages is found in the "CP Introduction" and "CP Directories" sections of this publication.

IBM 3850 MASS STORAGE SYSTEM (MSS) SUPPORT

New: Program and Documentation

Virtual machines operating CMS, OS/VS1, or OS/VS2 (MVS) can access mass storage volumes containing VM/370 minidisks or entire mass storage volumes dedicated to the virtual machine. These volumes will appear to the virtual machine as 3330 volumes and will be accessed using 3330 device support. CP controls unit allocation, volume mounting, and volume demounting. Virtual machines running OS/VS1 or OS/VS2 (MVS) and that contain MSS support can also access mass storage volumes using dedicated device support.

Communication with the Mass Storage Control (MSC) component of MSS is provided by service virtual machines operating either OS/VS1 or OS/VS2 (MVS). The VM/370 control program initiates volume mounts and demounts via intersystem communication with an application program (DMKMSS) running under either OS/VS1 or OS/VS2 (MVS). For information about the logic of the DMKMSS application program, see the VM/370 Service Routines Program Logic Manual, Order Number SY20-0882.

DIAGNOSE Code X'78' is added to communicate between a virtual machine and CP for MSS support. This information can be found in the "CP Introduction" section of this publication.

Appendix B is added to this publication to show flow diagrams of functions that utilize the MSS.

3800 PRINTING SUBSYSTEM SUPPORT

New: Program and Documentation

VM/370 supports the full facilites of the 3800 Printing Subsystem as a dedicated device. Limited 3800 printer support is provided by the VM/370 spooling facility.

DIAGNOSE Code X'74' is added to VM/370 to save or load a named system that contains control tables and control modules for the 3800 printer. The new NAME3800 macro instruction allows named systems to be specified in the same manner as the NAMENP macro instruction. This information is contained in the "CP Introduction", "CP Directories", and "CP Method of Operation and Program Organization" sections of this publication.

Two new utility programs, GENIMAGE and IMAGELIB construct or modify the tables and modules that control feature selections and printing on the 3800 printer. The IMAGELIB utility saves the control tables and control modules as a named system. Logic details on GENIMAGE and IMAGELIB are contained in the VM/370 Service Routines Program Logic Manual, Order Number SY20-0882.

CP ABEND CODES REMOVED

Changed: Documentation Only

To eliminate duplication, the CP Abend codes have been removed from this manual. They are found in the VM/370 System Message, Order Number GC20-1808.

Summary of Amendments
for SY20-0886-0
as updated by TNL SN25-0467
VM/370 Release 5 PLC 12

VARY PROCESSOR ONLINE/OFFLINE SUPPORT

New and Changed: Programming Support

VM/370 attached processor (AP) provides support to enhance the reliability, availability, and serviceability of the 158 and 168 attached processors, and the 158 and 168 asymmetric MP systems.

This support allows the attached processor to be taken offline to make needed repairs, and then be brought back online without affecting the main processor.

3340/3344 ALTERNATE TRACK SUPPORT

Changed: Program Support

Software error recovery procedures now provide for switching to an alternate track when an attempt to do I/O on a defective 3340 or 3344 track results in a track condition check. Logic affecting CP I/O, Diagnose I/O, and SIO issued from a virtual machine is changed. These logic changes are reflected in this publication.

CP Introduction

This part contains the following information:

- VM/370
- Program States
- Using Processor Resources
- Functional Information
- Performance Guidelines
- CP Interruption Handling

The VM/370 Control Program manages the resources of a single computer in such a manner that multiple computing systems appear to exist. 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 VM/370 directory. The virtual machine configuration includes counterparts of the components of a real IBM System/370:

- A virtual operator's console
- Virtual storage
- A virtual processor
- Virtual I/O devices

CP makes these components appear real to whichever operating system is controlling the work flow of the virtual machine.

The virtual machines operate concurrently via multiprogramming techniques. CP overlaps the idle time of one virtual machine with execution in another.

Each virtual machine is managed at two levels. 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.

VM/370 performs some functions differently when running in attached processor mode. For a description of the additional processing performed when in attached processor mode, see "The Attached Processor Environment" in this section.

Introduction to the VM/370 Control Program

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

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

The Control Program supervises 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. CP 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 to the virtual machine as a program interrupt. All virtual machine interrupts (including those caused by attempting privileged instructions) are first handled by CP, and are reflected to the virtual machine if an analogous interrupt would have occurred on a real machine.

VIRTUAL MACHINE TIME MANAGEMENT

The real processor simulates multiple virtual processors. Virtual machines that are executing in a conversational manner are given access to the real processor more frequently than those that are not; these conversational machines are assigned the smaller of two possible time slices. CP 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 processor utilization according to whether it is a conversational or nonconversational user of system resources.

A virtual machine can gain control of the processor only if it 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. The virtual machine cannot gain control of the real processor if it is waiting for a page of storage, if it is waiting for an input/output operation to be translated and started, or if it is waiting for a CP command to finish execution.

A virtual machine can be assigned a priority of execution. 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 is a parameter in the virtual machine's VM/370 directory entry. The system operator can reset the value with the privilege class A SET command.

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. You may redefine virtual storage size to any value that is a multiple of 4K and not greater than the maximum defined value. VM/370 implements this storage as virtual storage. The storage may appear as paged or unpaged to the virtual machine, depending upon whether or not the extended control mode option was specified for that virtual machine. This option is required if operating systems that control virtual storage, such as OS/VS1 or VM/370, are 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 updated by CP and reflect the allocation of virtual storage pages to blocks of real storage. These page and segment tables allow virtual storage addressing in a System/370 machine. Storage in the real machine is logically and physically divided into 4096-byte areas called page frames.

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 on virtual machines and from the pageable routines of CP compete for available page frames. When the number of page frames available for allocation falls below a threshold value, CP determines which virtual storage pages currently allocated to real storage are relatively inactive and initiates suitable page-out operations for them.

Inactive pages are kept on a direct access storage device. If an inactive page has been changed at some time during virtual machine

execution, CP 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 tell CP that the information from specific pages of virtual storage is no longer needed; CP then releases the areas of the paging devices which were assigned to hold the specified pages.

Paging is done on demand by CP. 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. CP makes no attempt to anticipate what pages might be required by a virtual machine. While a paging operation is performed for one virtual machine, another virtual machine can be executing. Any paging operation initiated by CP 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 kept. The virtual machine operating system is responsible for mapping the virtual storage created by it to the storage of the virtual machine. CP uses this set of tables in conjunction with the page and segment tables created for the virtual machine at logon 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.

Storage and Processor Utilization

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. CP will dynamically build up a set of reserved real storage page frames for this virtual machine during its execution until the maximum number "reserved" is reached. Since the pages of other virtual machines are not allocated from this reserved set, the effect is that most of the active pages of the selected virtual machine remain in real storage.

During CP system generation, the installation may specify an option called virtual=real. With this option, the virtual machine's storage is allocated directly from real storage at the time the virtual machine logs on (if it has the VIRT=REAL option in its directory). All pages except page zero are allocated to the corresponding real storage locations. In order to control the real computing system, real page zero must be controlled by CP. Consequently, the real storage size must be large enough to accommodate the CP nucleus, the entire virtual=real virtual machine, and the remaining pageable storage requirements of CP and the other virtual machines.

The virtual=real option improves performance in the selected virtual machine since it removes the need for CP paging operations for the selected virtual machine. The virtual=real option is necessary whenever programs that contain dynamically modified channel programs (excepting those of OS ISAM and OS/VS TCAM Level 5) are to execute under control of CP. For additional information on running systems with dynamically modified channel programs, see "Dynamically Modified Channel Programs" in VM/370 System Programmer's Guide.

A real disk device can be shared among multiple virtual machines. Virtual device sharing is specified in the VM/370 directory entry or by a user command. If specified by the user, an appropriate password must be supplied 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. CP checks each virtual machine input/output operation against the parameters in the virtual machine configuration to ensure device integrity.

Virtual Reserve/Release support can be used to further enhance device integrity for data on shared minidisks. Reserve/Release operation codes are simulated on a virtual basis for minidisks, including full-extent minidisks. For details on Reserve/Release support, refer to the topic "Reserve/Release," located under "Scheduling I/O Requests" in this section.

The virtual machine operating system is responsible for the operation of all virtual devices associated with it. These virtual devices may be defined in the VM/370 directory entry of the virtual machine, or they may be attached to (or detached from) the virtual machine's configuration, dynamically, for the duration of the terminal session. 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 CP 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 CP to translate the virtual specifications to real.

Virtual I/O can be initiated by either processor; however, all real I/O requests must be executed by the main processor, and all I/O interrupts must be received on the main processor (the processor with I/O capability). Any I/O requests by the attached processor (the processor without I/O capability) are transferred to the main processor.

In addition, the interrupts caused by the input/output operation are reflected to the virtual machine for its interpretation and processing. If input/output errors occur, CP records them but does not initiate error recovery operations. The virtual machine operating system must handle error recovery, but does not record the error (if SVC 76 is used).

Input/output operations initiated by CP for its own purposes (paging and spooling), are performed directly and are not subject to translation.

| See Appendix B of this volume for an explanation of additional
| processing when the virtual I/O request results in a real I/O request to
| an MSS 3330V volume.

Dedicated Channels

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

By specifying a dedicated channel (or channels) for a virtual machine via the Class B ATTACH CHANNEL command, the CP channel scheduling function is bypassed for that virtual machine. A virtual machine assigned a dedicated channel has that channel and all of its devices for its own exclusive use. CP translates the virtual storage locations specified in channel commands to real locations and performs any necessary paging operations, but does not perform any device address translations. The virtual device addresses on the dedicated channel must match the real device addresses; thus, a minidisk cannot be used.

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.

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

Spooling operations cease if the direct access storage space assigned to spooling is exhausted, and the virtual unit record devices 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. 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.

CP spooling includes many desirable options for the virtual machine user and the real machine operator. These options 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. Each output spool file has, associated with it, a 136-byte area known as the spool file tag. The information contained in this area and its syntax are determined by the originator and receiver of the file. For example, whenever an output spool file is destined for transmission to a remote location via the Remote Spooling Communications Subsystem, RSCS expects to find the destination identification in the file tag. Tag data is set, changed, and queried using the CP TAG command.

It is possible to spool terminal input and output. All data sent to the terminal, whether it be from the virtual machine, the control program or the virtual machine operator, can be spooled. Spooling is particularly desirable when a virtual machine is run with its console disconnected. Console spooling is usually started via the command

```
SPOOL CONSOLE START
```

An exception to this is when a system operator logs on using a graphics device. In this instance, console spooling is automatically started and continues in effect even if the system operator should disconnect from the graphics device and log on to a nongraphic device. In order to stop automatic console spooling, the system operator must issue the command

```
SPOOL CONSOLE STOP
```

SPOOL FILE RECOVERY

If the system should suffer an abnormal termination, there are three degrees of recovery for the system spool files; warm start (WARM), checkpoint start (CKPT), and force start (FORCE). Warm start is automatically invoked if SET DUMP AUTO is in effect. Otherwise, the choice of recovery method is selected when the following message is issued;

```
hh:mm:ss START ((COLD|WARM|CKPT|FORCE)(DRAIN))|(SHUTDOWN):
```

Note that a cold (COLD) start does not recover any spool files.

Warm Start

After a system failure, the warm start procedure copies spool file, accounting, and system message data to warm start cylinders on an auxiliary DASD. When the system is reloaded, this information is retrieved and the spool file chains and other system data are restored to their original status. If the warm start procedure cannot be implemented because certain required areas of storage are invalid, the operator is notified to take other recovery procedures.

Checkpoint Start

Any new or revised status of spool file blocks, spooling devices, and spool hold queue blocks is dynamically copied to checkpoint cylinders on an auxiliary DASD as they occur. When a checkpoint (CKPT) start is requested, this is the information that is used to recreate the spool file chains. It differs from warm start data in that only spool file data is restored; accounting and system messages information is not recovered. Also, the order of spool files on any particular restored chain is not the original sequence but a random one.

Force Start

A force start is required when checkpoint start encounters I/O errors while reading files, or invalid data. The procedure is the same as for checkpoint start except that unreadable or invalid files are bypassed.

CP COMMANDS

The CP commands allow you to control the 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 (for 3066 and 3270 terminals, the ENTER key is used); it can be restarted by entering the appropriate CP command. External, attention, and device ready interrupts can be simulated on 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 a single-instruction mode. Commands are available to invoke the spooling and disk sharing functions of CP.

CP commands are classified by privilege classes. The VM/370 directory entry for each user assigns one or more privilege classes. The classes are primary system operator (class A), system resource operator (class B), system programmer (class C), spooling operator (class D), system analyst (class E), service representative (class F), and general user (class G). Commands in the system analyst class may be used to inspect real storage locations, 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 commands related to virtual machine performance options, such as assigning a set of reserved page frames to a selected virtual machine. For descriptions of all the CP commands, see the VM/370 CP Command Reference for General Users and the VM/370 Operator's Guide.

Program States

When instructions in the Control Program are being executed, the real computer is in the supervisor state; at all other times, when running virtual machines, the real computer is in the problem state. Therefore, privileged instructions cannot be executed by the virtual machine. Programs running on a virtual machine can issue privileged instructions; but such an instruction either (1) causes an interruption that is handled by the Control Program, or (2) is intercepted and handled by the processor, if the virtual machine assist feature or VM/370 Extended Control-Program Support is enabled and supports that instruction. CP examines the operating status of the virtual machine PSW. If the virtual machine indicates that it is functioning in supervisor mode, the privileged instruction is simulated according to its type. If the virtual machine is in problem mode, the privileged interrupt is reflected to the virtual machine.

Only the Control Program may operate in the supervisor state on the real machine. All programs other than CP operate in the problem state on the real machine. All user interrupts, including those caused by attempted privileged operations, are handled by either the control program or the processor (if the virtual machine assist feature or VM/370 Extended Control-Program Support is available). Only those interrupts that the user program would expect from a real machine are reflected to it. A problem program will execute on the virtual machine in a manner identical to its execution on a real System/370 processor, as long as it does not violate the CP restrictions. See VM/370 System Messages for a list of the restrictions.

Using Processor Resources

CP allocates the processor resource to virtual machines according to their operating characteristics, priority, and the system resources available.

Virtual machines are dynamically categorized at the end of each time slice as interactive or noninteractive, depending upon the frequency of operations to or from either the virtual system console or a terminal controlled by the virtual machine.

Virtual machines are dispatched from one of two queues, called Queue 1 and Queue 2. In order to be dispatched from either queue, a virtual machine must be considered executable (that is, not waiting for some activity or for some other system resource). Virtual machines are not considered dispatchable if the virtual machine:

- Enters a virtual wait state after an I/O operation has begun.
- Is waiting for a page frame of real storage.
- Is waiting for an I/O operation to be translated by CP and started.
- Is waiting for CP to simulate its privileged instructions.
- Is waiting for a CP console function to be performed.

Queue 1

Virtual machines in Queue 1 (Q1) are considered conversational or interactive users, and enter this queue when an interrupt from a terminal is reflected to the virtual machine. Users are considered for dispatching from this queue on a first-in-first-out (FIFO) basis. When a virtual machine uses more than a certain amount of processor time without entering a virtual wait state, that user is placed in Queue 2.

Virtual machines are dropped from Q1 when they complete their time slice of processor usage, and are placed in an "eligible list". Virtual machines entering CP command mode are also dropped from Q1. When the virtual machine becomes executable again (returns to execution mode) it is placed at the bottom of Q1.

Queue 2

Virtual machines in Queue 2 (Q2) are considered noninteractive users. Users are selected to enter Q2 from a list of eligible virtual machines (the "eligible list"). The list of eligible virtual machines is sorted on a FIFO basis within user priority (normally defined in the user record in the VM/370 directory, but may be altered by the system operator).

Usually, a virtual machine is selected to enter Q2 only if its "working set" is not greater than the number of real page frames available for allocation at the time. The working set of a virtual machine is calculated and saved each time a user is dropped from Q2 and is based on the number of virtual pages referred to by the virtual machine during its stay in Q2, and the number of its virtual pages that are resident in real storage at the time it is dropped from the queue.

If the calculated working set of the highest priority virtual machine in the eligible list is greater than the number of page frames available for allocation, then 75 percent of the working set for that virtual machine is calculated. If the pages required for 75 percent of the working set are available, the virtual machine is placed on Q2. Otherwise, the virtual machine remains on the eligible list until there are no other users on Q1 or Q2.

Executable virtual machines are sorted by "dispatching priority". This priority is calculated each time a user is dropped from a queue and is the ratio of processor time used while in the queue to elapsed time in the queue. Infrequent processor users are placed at the top of the list and are followed by more frequent processor users. When a nonexecutable user becomes executable, he is placed on the queue based on his dispatching priority.

When a virtual machine completes its time slice of processor usage, it is dropped from Q2 and placed in the eligible list by user priority. When a user request in Q2 enters CP command mode, it is removed from Q2. When the request becomes executable (returns to virtual machine execution mode), it is placed in the eligible list based on user priority.

If a user's virtual machine is not in Q1 or Q2, it is because:

- The virtual machine is on the "eligible list," waiting to be put on Q2

  -- or --

- The virtual machine execution is suspended because the user is in CP mode executing CP commands

To leave CP mode and return his virtual machine to the "eligible list" for Q2, the user can issue one of the CP commands that transfer control to the virtual machine operating system for execution (for example, BEGIN, IPL, EXTERNAL, and RESTART).

In CP, interactive users (Q1), if any, are considered for dispatching before noninteractive users (Q2). This means that CMS users entering commands that do not involve disk or tape I/O operations should get fast responses from the VM/370 system even with a large number of active users.

An installation may choose to override the CP scheduling and dispatching scheme and force allocation of the processor resource to a specified user, regardless of its priority or operating characteristics. The favored execution facility allows an installation to:

1. Specify that one particular virtual machine is to receive up to a specified percentage of processor time.

2. Specify that any number of virtual machines are to remain in the queues at all times. Assignment of the favored execution option is discussed in the "Preferred Virtual Machines" section.

Functional Information

The functional diagrams that follow describe the program logic associated with various control program functions. Not all CP functions are described. These functional diagrams are meant to describe the CP functions about which you may want more detailed information if you are debugging, modifying, or updating CP.

Figure 1 describes CP initialization process.

Figures 2 and 3 describe the real and virtual I/O control blocks used by CP in its I/O control.

Figures 4, 5, and 6 show how CP handles SVC, external, and program interrupts.

The CP paging function is described in Figure 7.

The CP spooling function (both virtual and real) is described in Figures 8 and 9.

Figure 10 shows how virtual tracing is performed.

Figure 11 shows the steps involved in translating a virtual address to a real address and gives an example of address translation.

The functional information contained in these diagrams is intended for system programmers and IBM Field Engineering program support representatives.

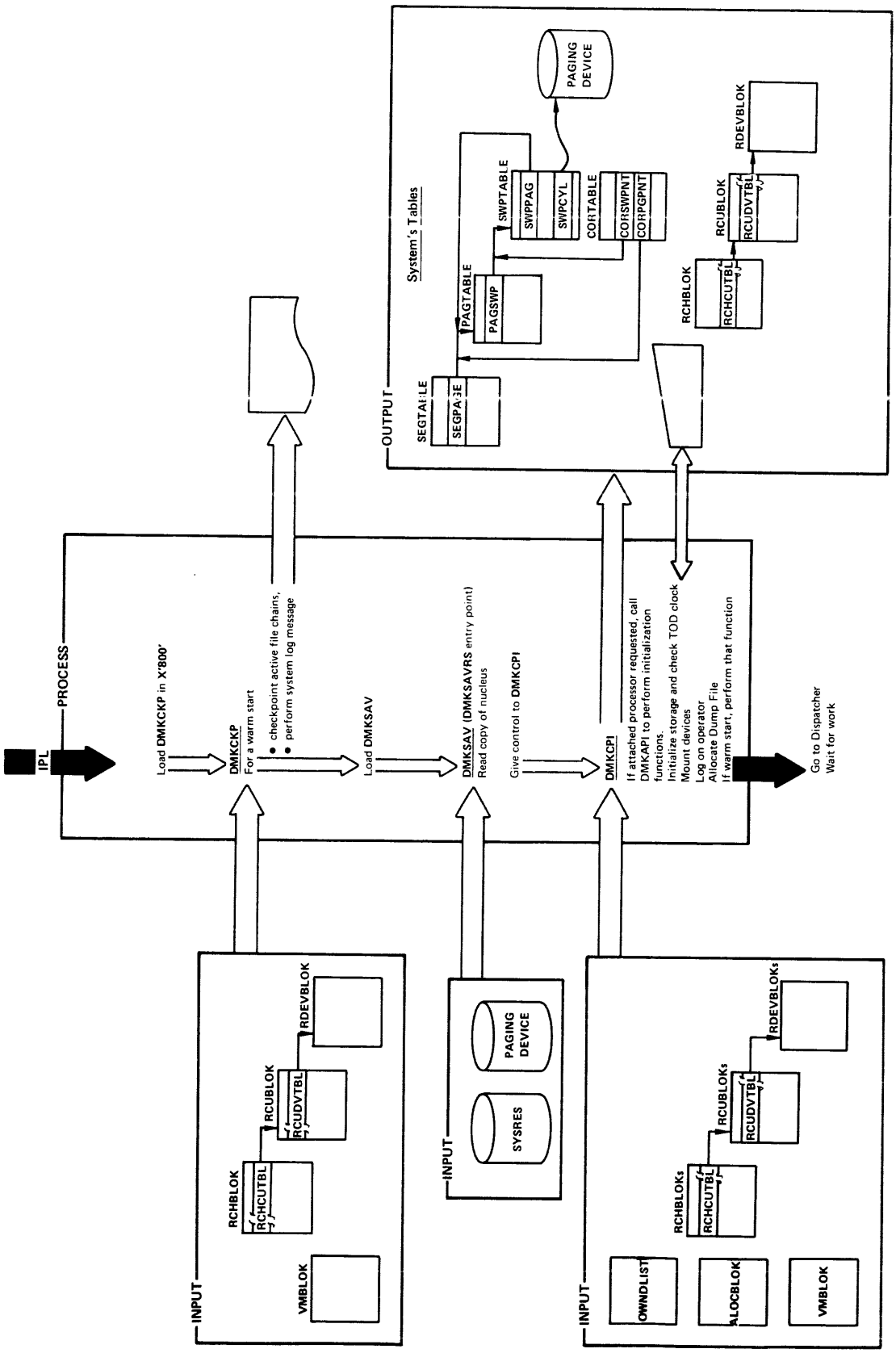


Figure 1. CP Initialization

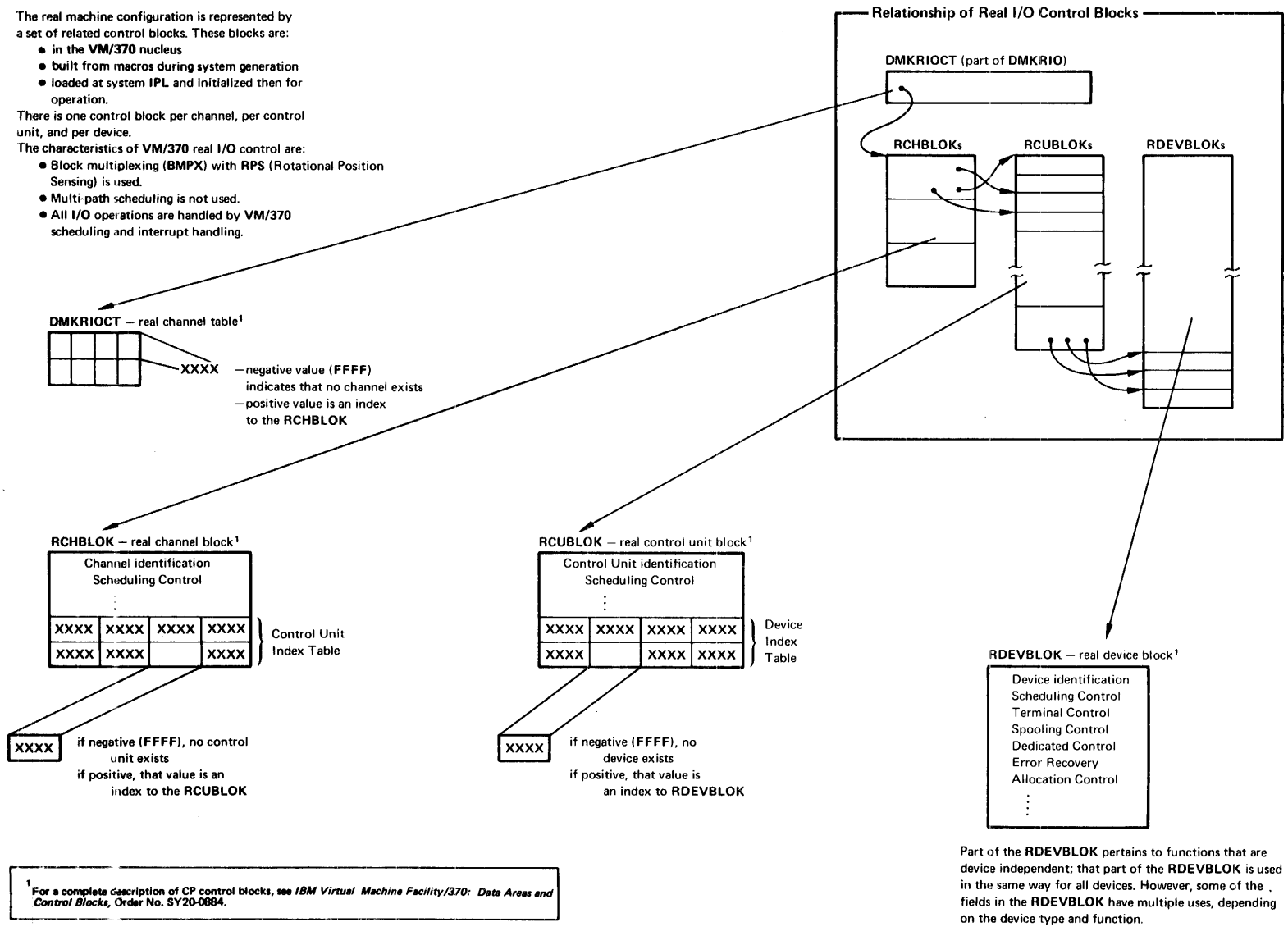


Figure 2. Real I/O Control Blocks

Figure 3. Virtual I/O Control Blocks

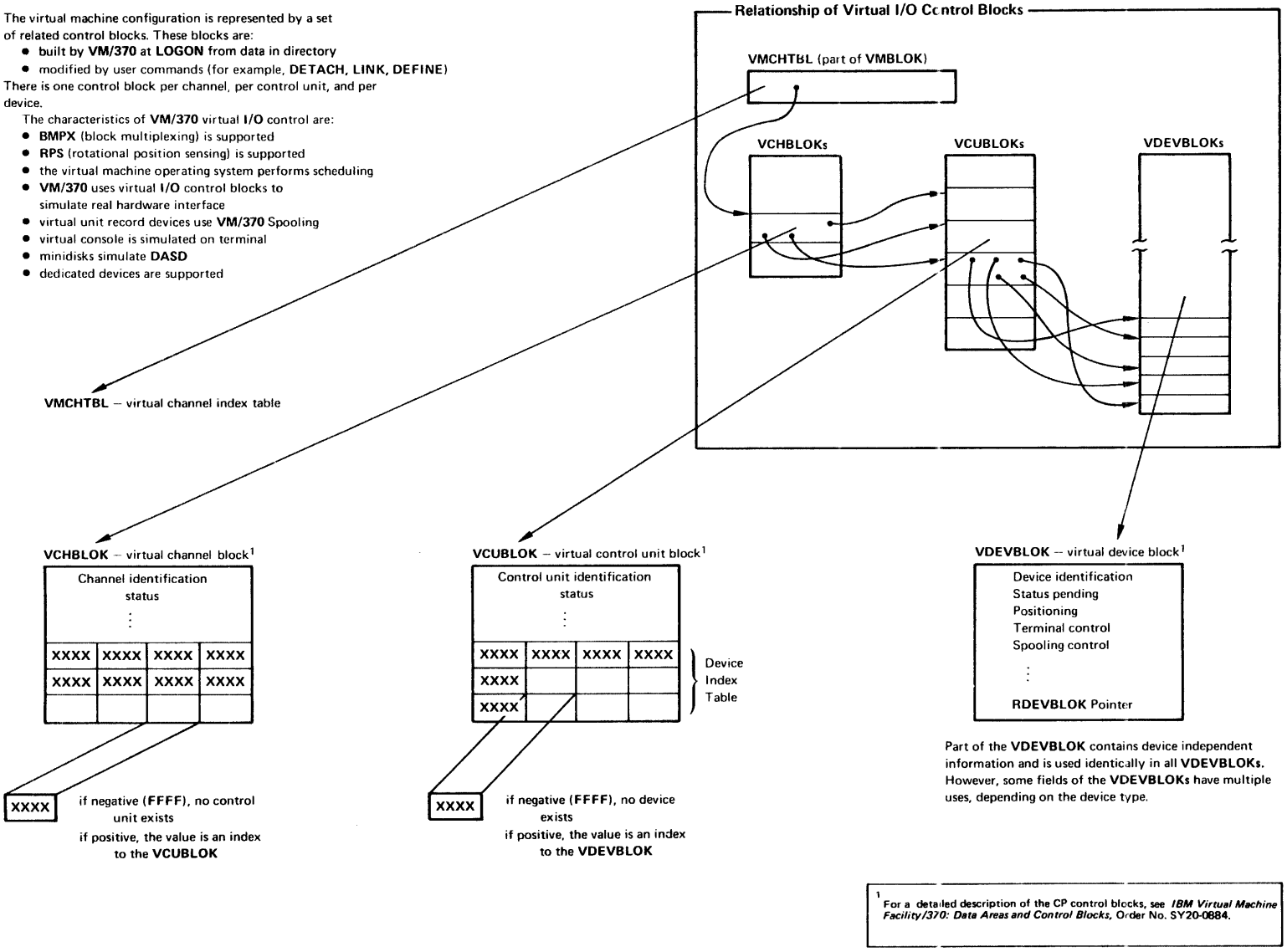
The virtual machine configuration is represented by a set of related control blocks. These blocks are:

- built by **VM/370** at **LOGON** from data in directory
- modified by user commands (for example, **DETACH, LINK, DEFINE**)

There is one control block per channel, per control unit, and per device.

The characteristics of **VM/370** virtual **I/O** control are:

- **BMPX** (block multiplexing) is supported
- **RPS** (rotational position sensing) is supported
- the virtual machine operating system performs scheduling
- **VM/370** uses virtual **I/O** control blocks to simulate real hardware interface
- virtual unit record devices use **VM/370** Spooling
- virtual console is simulated on terminal
- minidisks simulate **DASD**
- dedicated devices are supported



Part of the **VDEVBLOK** contains device independent information and is used identically in all **VDEVBLOKs**. However, some fields of the **VDEVBLOKs** have multiple uses, depending on the device type.

[1] For a detailed description of the CP control blocks, see *IBM Virtual Machine Facility/370: Data Areas and Control Blocks*, Order No. SY20-0884.

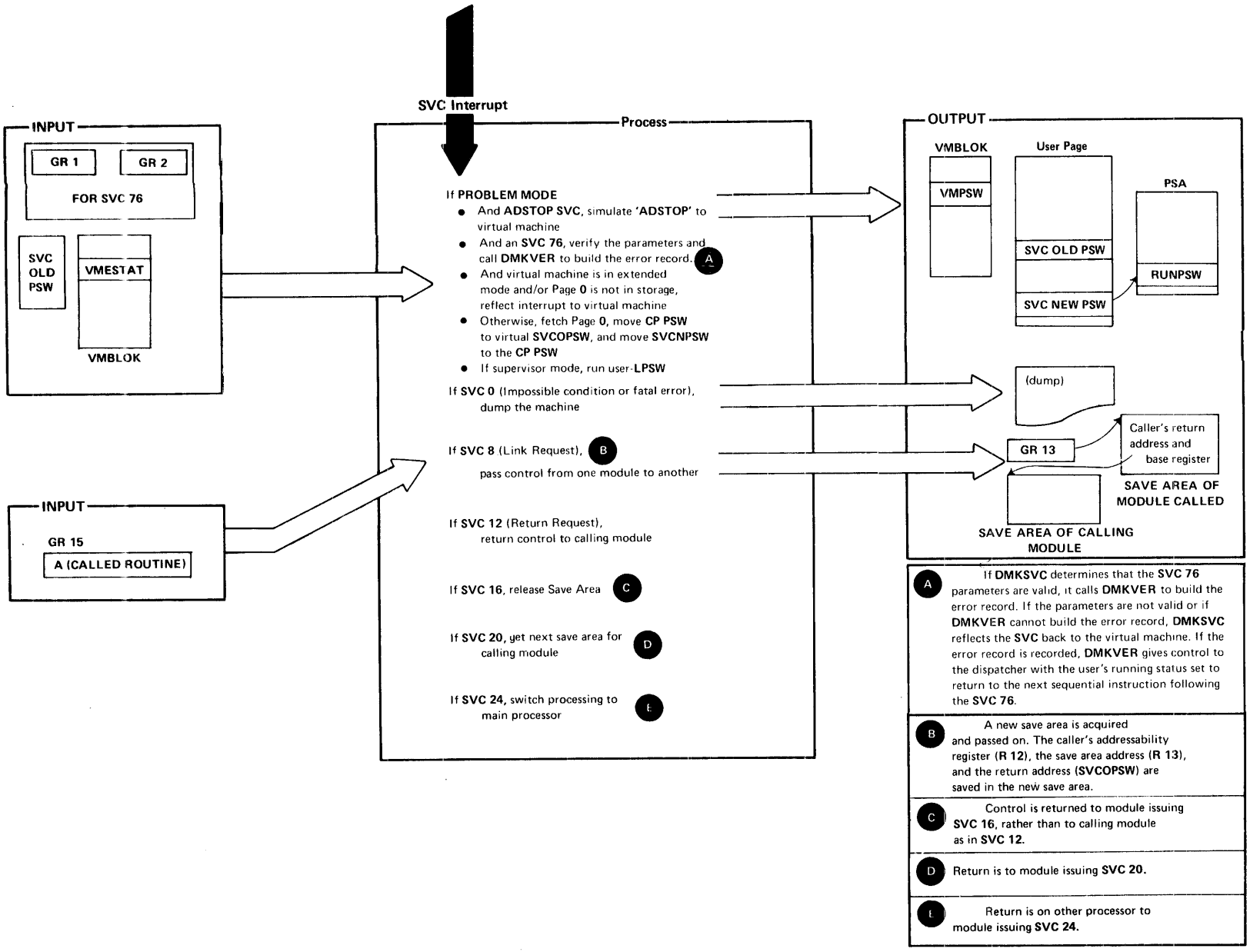
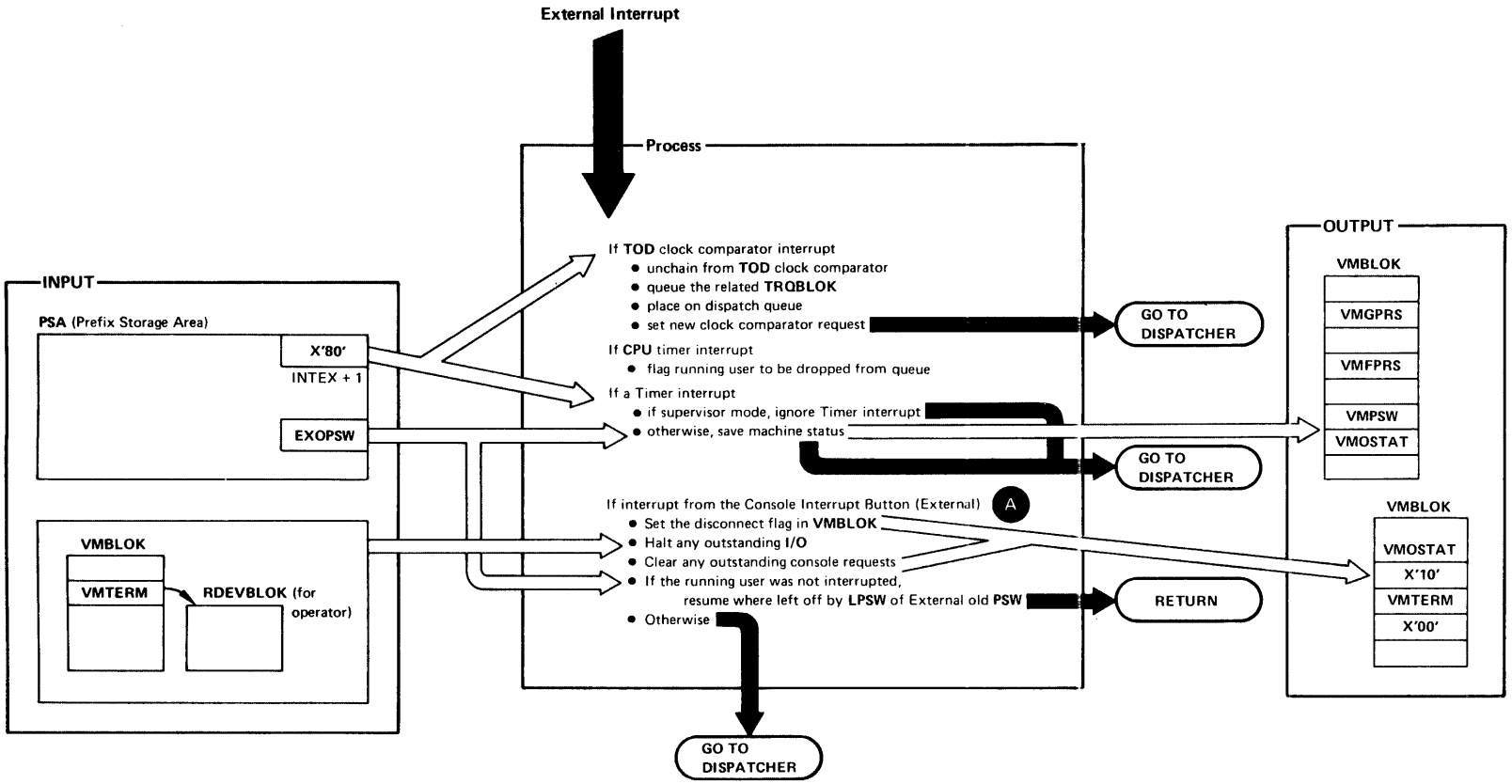


Figure 4. SVC Interrupt Handling

Figure 5. External Interrupt Handling



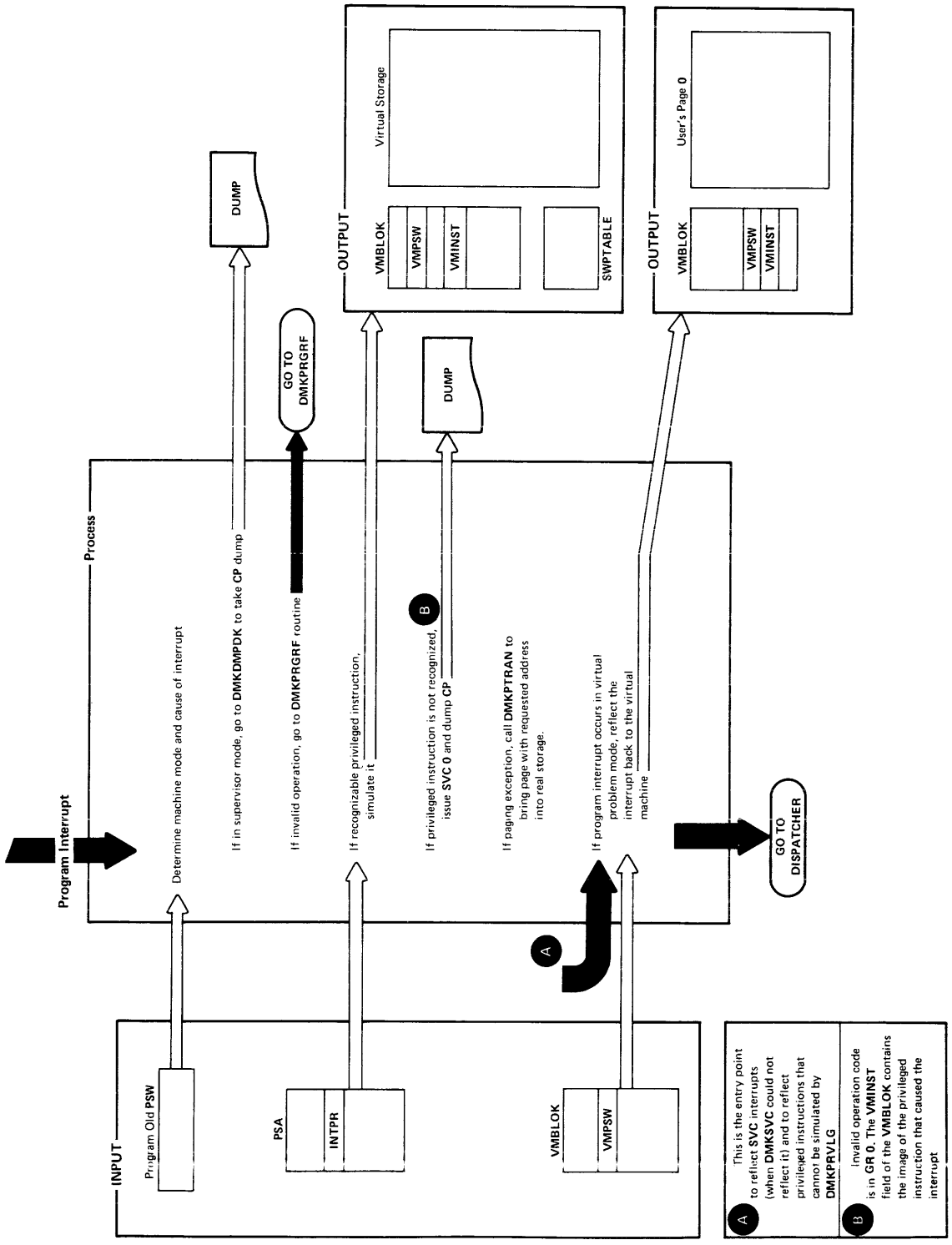
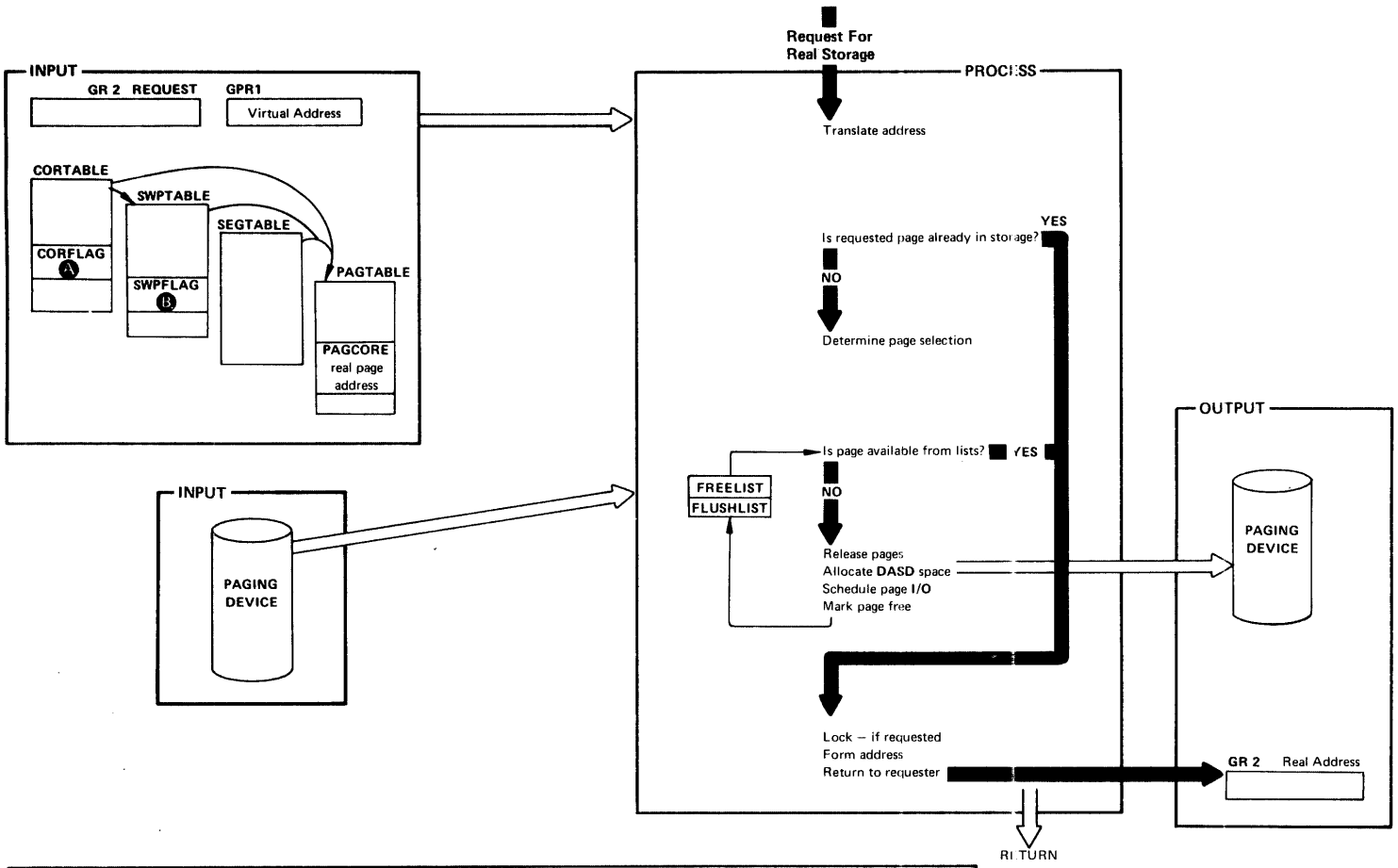


Figure 6. Program Interrupt Handling

Figure 7. Paging



A Bits defined for CORFLAG			
CORIOLCK	EQU	X'80'	Page locked for I/O
CORCFLCK	EQU	X'40'	Page locked by console function
CORFLUSH	EQU	X'20'	Page is in flush list
CORFREE	EQU	X'10'	Page is in free list
CORSHARE	EQU	X'08'	Page is shared
CORRSV	EQU	X'04'	Page is reserved
CORDISA	EQU	X'01'	Page disabled – not available

B Bits defined for SWPFLAG			
SWPTRANS	EQU	X'80'	Page in transit
SWPRECMP	EQU	X'40'	Page permanently assigned
SWPALLOC	EQU	X'20'	Page enqueued for allocation
SWPSHR	EQU	X'10'	Page shared
SWPREF1	EQU	X'08'	1st half page referenced
SWPCHG1	EQU	X'04'	1st half page changed
SWPREF2	EQU	X'02'	2nd half page referenced
SWPCHG2	EQU	X'01'	2nd half page changed

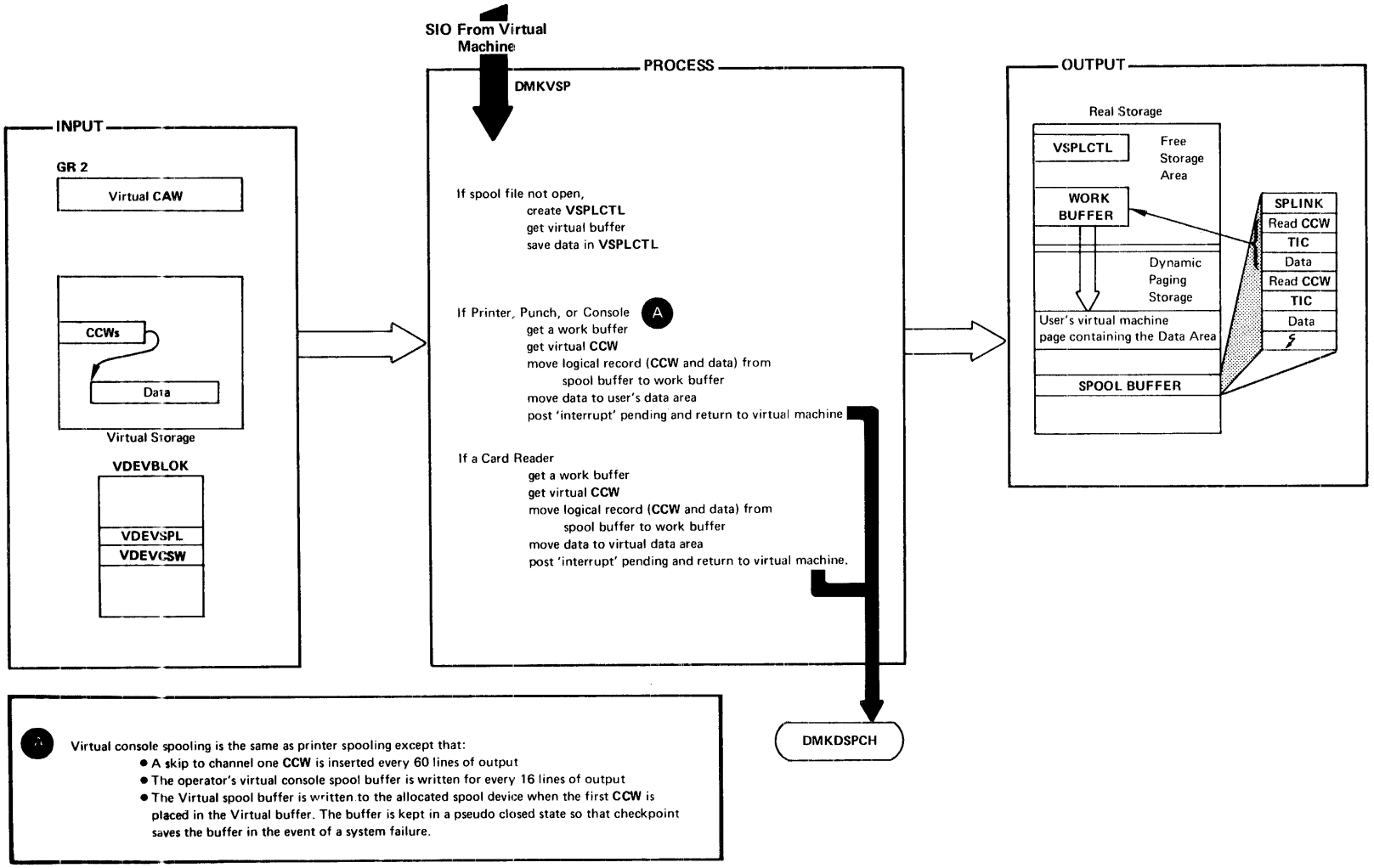


Figure 8. Virtual Spooling

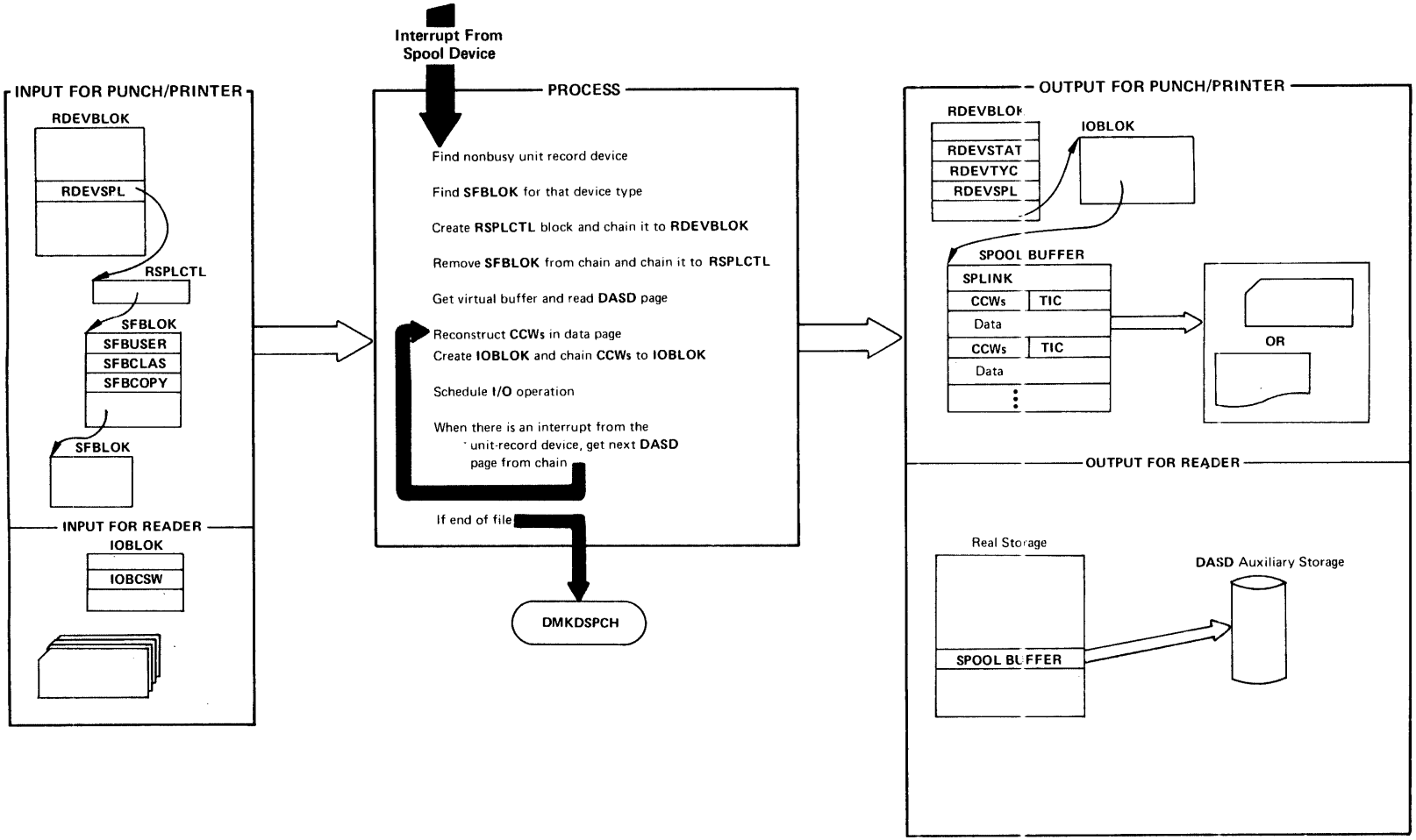


Figure 9. Real Spooling

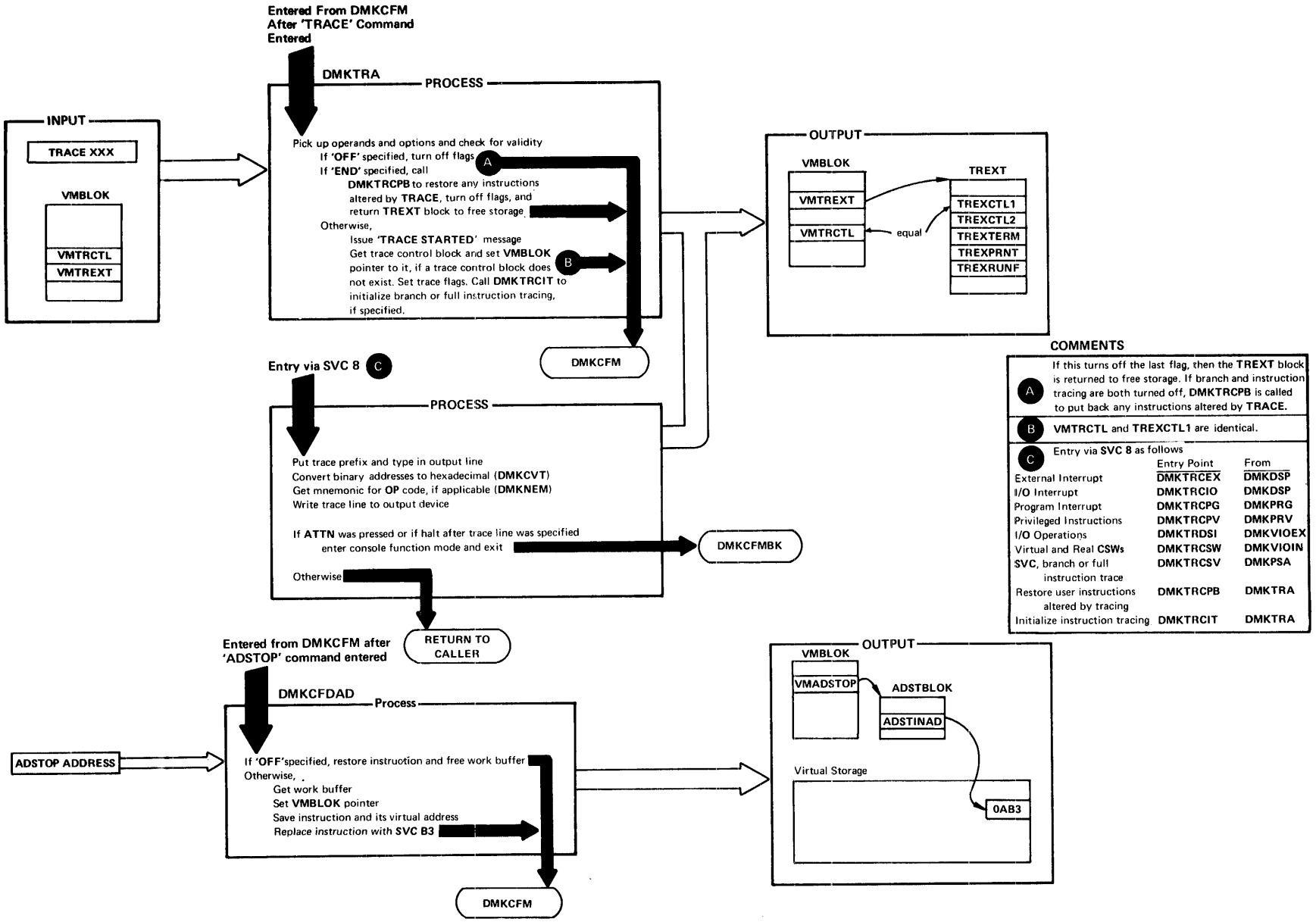


Figure 10. Virtual Tracing

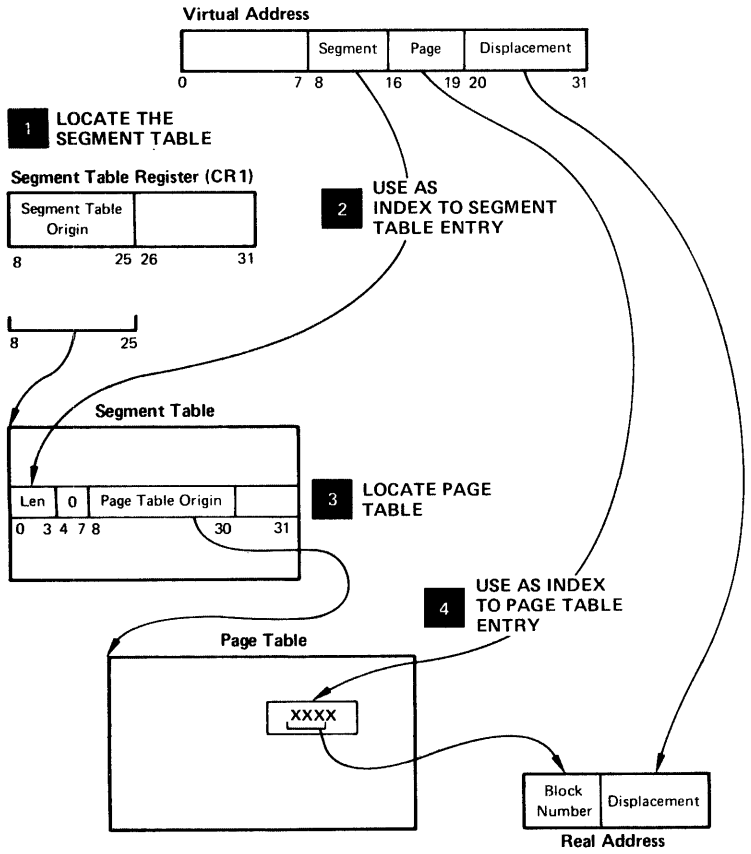
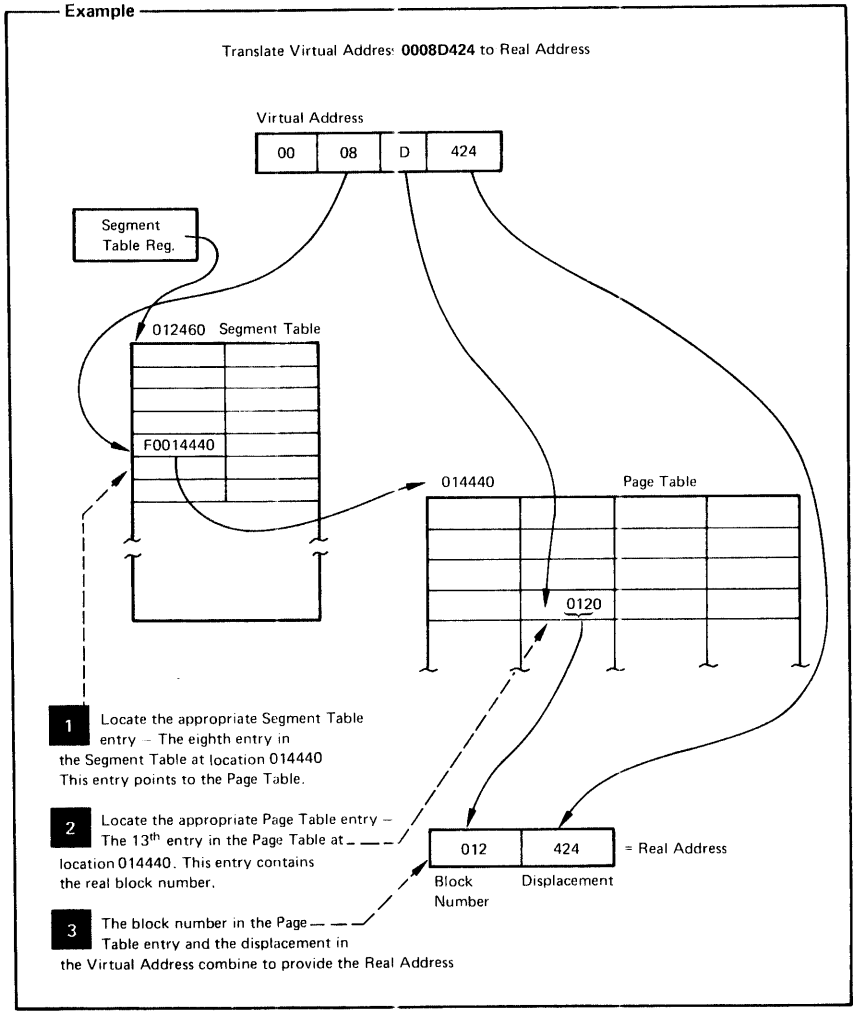


Figure 11. Virtual-to-Real Address Translation



Performance Guidelines

General Information

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

- The System/370 model used.
- The total number of virtual machines executing.
- The type of work being done by each virtual machine.
- The speed, capacity, and number of the paging devices.
- The amount of real storage available.
- The degree of channel and control unit contention, as well as arm contention, affecting the paging device.
- The type and number of VM/370 performance options in use by one or more virtual machines.
- The degree of access to MSS 3330V volume.

Performance of any virtual machine may be improved up to some limit by the choice of hardware, operating system, and VM/370 options. The topics discussed in this section address:

1. The performance options available in VM/370 to improve the performance of a particular virtual machine.
2. The system options and operational characteristics of operating systems running in virtual machines that will affect their execution in the virtual machine environment.

The performance of a specific virtual machine may never equal that of the same operating system running standalone on the same System/370, but the total throughput obtained in the virtual machine environment may equal or better that obtained on a real machine.

When executing in a virtual machine, any function that cannot be performed wholly by the hardware causes some degree of degradation in the virtual machine's performance. As the control program for the real machine, CP initially processes all real interrupts. A virtual machine operating system's instructions are always executed in problem state. Any privileged instruction issued by the virtual machine causes a real privileged instruction exception interruption. The amount of work to be done by CP to analyze and handle a virtual machine-initiated interrupt depends upon the type and complexity of the interrupt.

The simulation effort required of CP may be trivial, as for a supervisor call (SVC) interrupt (which is generally reflected back to the virtual machine), or may be more complex, as in the case of a Start I/O (SIO) interrupt, which initiates extensive CP processing.

When planning for the virtual machine environment, consideration should be given to the number and type of privileged instructions to be executed by the virtual machines. Any reduction in the number of privileged instructions issued by the virtual machine's operating system will reduce the amount of extra work CP must do to support the machine.

Virtual Machine I/O

To support I/O processing in a virtual machine, CP must translate all virtual machine channel command word (CCW) sequences to refer to real storage and real devices and, in the case of minidisks, real cylinders. When a virtual machine issues an SIO, CP must:

1. Intercept the virtual machine SIO interrupt.
2. Allocate real storage space to hold the real CCW list to be created.
3. Translate the virtual device addresses referred to in the virtual CCWs to real addresses.
4. Page into real storage and lock, for the duration of the I/O operation, all virtual storage pages required to support the I/O operation.
5. Generate a new CCW sequence building a Channel Indirect Data Address list if the real storage locations cross page boundaries.
6. Schedule the I/O request.
7. Present the SIO condition code to the virtual machine.
8. Intercept, retranslate, and present the channel end and device end interrupts to the appropriate virtual machine, where they must then be processed by the virtual machine operating system.

CP's handling of SIOs for virtual machines can be one of the most significant causes of reduced performance in virtual machines.

The number of SIO operations required by a virtual machine can be significantly reduced in several ways:

- Use of large blocking factors (of up to 4096 bytes) for user data sets to reduce the total number of SIOs needed.
- Use of preallocated data sets.
- Use of virtual machine operating system options (such as chained scheduling in OS) that reduce the number of SIO instructions.
- Substitution of a faster resource (virtual storage) for I/O operations, by building small temporary data sets in virtual storage rather than using an I/O device.

Frequently, there can be a performance gain when CP paging is substituted for virtual machine I/O operations. The performance of an operating system such as OS can be improved by specifying as resident as many frequently used OS functions (transient subroutines, ISAM indexes, and so forth) as are possible. In this way, paging I/O is substituted for virtual machine-initiated I/O. In this case, the only work to be done by CP is to place into real storage the page that contains the desired routine or data.

Three CP performance options are available to reduce the CP overhead associated with virtual machine I/O instructions or other privileged instructions used by the virtual machine's I/O Supervisor:

1. The virtual=real option removes the need for CP to perform storage reference translation and paging before each I/O operation for a specific virtual machine.

2. The virtual machine assist reduces the real supervisor state time used by VM/370. See VM/370 Planning and System Generation Guide for a list of the processors on which it is available.

3. VM/370 Extended Control-Program Support further reduces the real supervisor state time used by VM/370. See VM/370 Planning and System Generation Guide for a list of the processors on which it is available.

Assignment and use of these options is discussed in "Preferred Virtual Machines."

Paging Considerations

When virtual machines refer to virtual storage addresses that are not currently in real storage, they cause a paging exception and the associated CP paging activity.

The addressing characteristics of programs executing in virtual storage have a significant effect on the number of page exceptions experienced by that virtual machine. Routines that have widely scattered storage reference tend to increase the paging load of a particular virtual machine. When possible, modules of code that are dependent upon each other should be located in the same page. Reference tables, constants, and literals should also be located near the routines that use them. Exception or error routines that are infrequently used should not be placed within main routines, but located elsewhere.

When an available page of virtual storage contains only reenterable code, paging activity can be reduced, since the page, although referred to, is never changed, and thus does not cause a write operation to the paging device. The first copy of that page is written on the paging device when that frame is needed for some other more active page. Only inactive pages that have changed must be paged out.

Virtual machines that reduce their paging activity by controlling their use of addressable space improve resource management for that virtual machine, the VM/370 system, and all other virtual machines. The total paging load that must be handled by CP is reduced, and more time is available for productive virtual machine use.

Additional dynamic paging storage may be gained by controlling free storage allocation. The amount of free storage allocated at VM/370 initialization time can be controlled by the installation. When the System is being generated, the FREE operand of the SYSCOR macro statement may be used to specify the number of free storage pages to be allocated at system load time.

If, at IPL time, the amount of storage that these pages represent is greater than 25 percent of the VM/370 storage size (not including the V=R area, if any), a default number of pages is used. The default value is 3 pages for the first 256K bytes of storage plus 1 page for each additional 64K bytes (not including the V=R size, if any).

The SYSCOR macro definition can be found in VM/370 Planning and System Generation Guide.

CP provides three performance options, locked pages, reserved page frames, and a virtual=real area, to reduce the paging requirements of virtual machines. Generally, these facilities require some dedication of real storage to the chosen virtual machine and, therefore, improve its performance at the expense of other virtual machines.

LOCKED PAGES OPTION

The LOCK command, which is available to the system operator (with privilege class A), can be used to permanently fix or lock specific user pages of virtual storage into real storage. In so doing, all paging I/O for these page frames is eliminated.

Since this facility reduces total real storage resources (real page frames) that are available to support other virtual machines, only frequently used pages should be locked into real storage. Since page zero (the first 4096 bytes) of a virtual machine storage is referred to and changed frequently (for example, whenever a virtual machine interrupt occurs or when a CSW is stored), it should be the first page of a particular virtual machine that an installation considers locking. The virtual machine interrupt handler pages might also be considered good candidates for locking.

Other pages to be locked depend upon the work being done by the particular virtual machine and its usage of virtual storage.

The normal CP paging mechanism selects unreferenced page frames in real storage for replacement by active pages. Page frames belonging to inactive virtual machines will all eventually be selected and paged out if the real storage frames are needed to support active virtual machine pages.

When virtual machine activity is initiated on an infrequent or irregular basis, such as from a remote terminal in a teleprocessing inquiry system, some or all of its virtual storage may have been paged out before the time the virtual machine must begin processing. Some pages will then have to be paged in so that the virtual machine can respond to the teleprocessing request compared with running the same teleprocessing program on a real machine. This paging activity may cause an increase in the time required to respond to the request compared with running the teleprocessing program on a real machine. Further response time is variable, depending upon the number of paging operations that must occur.

Locking specific pages of the virtual machine's program into real storage may ease this problem, but it is not always easy nor possible to identify which specific pages will always be required.

Once a page is locked, it remains locked until either the user logs off or the system operator (privilege class A) issues the UNLOCK command for that page. If the "locked pages" option is in effect and the user loads his system again (via IPL) or loads another system, the locked pages are refreshed and the virtual machine's locked pages are unlocked by the system. The SYSTEM CLEAR command, when invoked, clears virtual machine storage, including the user's locked pages.

Note: In attached processor mode, no shared pages are locked. If the system operator attempts to lock a shared page or an address range containing one or more shared pages, he will receive the message

DMKCPV165I PAGE (hexloc) NOT LOCKED, SHARED PAGE

for each of the shared pages within the range.

RESERVED PAGE FRAMES OPTION

A more flexible approach than locked pages is the reserved page frames option. This option provides a specified virtual machine with an essentially private set of real page frames, the number of frames being designated by the system operator, when he issues the CP SET RESERVE command line. Pages will not be locked into these frames. They can be paged out, but only for other active pages of the same virtual machine. When a temporarily inactive virtual machine having this option is reactivated, these page frames are immediately available. If the program code or data required to satisfy the request was in real storage at the time the virtual machine became inactive, no paging activity is required for the virtual machine to respond.

This option is usually more efficient than locked pages in that the pages that remain in real storage are those pages with the greatest amount of activity at that moment, as determined automatically by the system. Although multiple virtual machines may use the LOCK option, only one virtual machine at a time may have the reserved page frames option active. Assignment of this option is discussed further in "Preferred Virtual Machines."

The reserved page frames option provides performance that is generally consistent from run to run with regard to paging activity. This can be especially valuable for production-oriented virtual machines with critical schedules, or those running teleprocessing applications where response times must be kept as short as possible.

VIRTUAL=REAL OPTION

The VM/370 virtual=real option eliminates CP paging for the selected virtual machine. All pages of virtual machine storage, except page zero, are locked in the real storage locations they would use on a real computer. CP controls real page zero, but the remainder of the CP nucleus is relocated and placed beyond the virtual=real machine in real storage. This option is discussed in more detail in "Preferred Virtual Machines."

Since the entire address space required by the virtual machine is locked, these page frames are not available for use by other virtual machines except when the virtual=real machine is not logged on. This option often increases the paging activity for other virtual machine users, and in some cases for VM/370. (Paging activity on the system may increase substantially, since all other virtual machine storage requirements must be managed with fewer remaining real page frames.)

The virtual=real option may be desirable or mandatory in certain situations. The virtual=real option is desirable when running a virtual machine operating system (like DOS/VS or OS/VS) that performs paging of its own because the possibility of double paging is eliminated. The option must be used to allow programs that execute self-modifying channel programs or have a certain degree of hardware timing dependencies to run under VM/370.

Preferred Virtual Machine Options

VM/370 provides seven functions that create a special virtual machine environment:

1. Favored execution
2. Priority
3. Reserved page frames
4. Virtual=real option
5. Affinity
6. Virtual machine assist
7. Extended Control-Program Support

The first five functions are designed to improve the performance of a selected virtual machine; the last two functions improve the performance of VM/370. Although each of the first five functions could be applied to a different virtual machine, usually they are applied to only one if optimum performance is required for that one specific virtual machine. The sixth and seventh functions can be applied to as many virtual machines as desired.

FAVORED EXECUTION

The favored execution options allow an installation to modify the normal scheduling algorithms and force the system to devote more of its processor resources to a given virtual machine than would ordinarily be the case. The options provided are:

1. The basic favored execution option.

2. The favored execution percentage option.

The basic favored execution option means that the virtual machine so designated is not to be dropped from the active (in queue) subset by the scheduler, unless it becomes nonexecutable. When the virtual machine is executable, it is to be placed in the dispatchable list at its normal priority position. However, any active virtual machine represents either an explicit or implicit commitment of main storage. An explicit storage commitment can be specified by either the virtual=real option or the reserved page frames option. An implicit commitment exists if neither of these options is specified, and the scheduler recomputes the virtual machine's projected work-set at what it would normally have been at queue-drop time. Multiple virtual machines can have the basic favored execution option set. However, if their combined main storage requirements exceed the system's capacity, performance can suffer because of thrashing.

If the favored task is highly compute bound and must compete for the processor with many other tasks of the same type, an installation can define the processor allocation to be made. In this case, the favored execution percentage option can be selected for one virtual machine.

This option specifies that the selected virtual machine, in addition to remaining in queue, is guaranteed a specified minimum percentage of the total processor time if it can use it. The favored execution option can only be invoked by a system operator with command privilege class A. The format of the command is as follows:

```
                    ┌    ┐
SET FAVORED userid  |nn  |
                    └    ┘
```

where:

userid identifies the virtual machine to receive favored execution status.

nn is any value from 1 through 99 and specifies the percentage of the in-queue time slice that is guaranteed to this virtual machine.

OFF specifies that the virtual machine is to be removed from favored execution status.

The percentage option of the SET FAVORED command is administered as follows:

1. The in-queue time slice is multiplied by the specified percentage to arrive at the virtual machine's guaranteed processor time.
2. The favored virtual machine, when it is executable, is always placed at the top of the dispatchable list until it has obtained its guaranteed processor time.
3. If the virtual machine obtains its guaranteed processor time before the end of its in-queue time slice, it is placed in the dispatchable list according to its calculated dispatching priority.
4. In either case (2 or 3), at the end of the in-queue time slice the guarantee is recomputed as in step 1 and the process is repeated.

Whether or not a percentage is specified, a virtual machine with the favored execution option active is kept in the dispatching queues except under the following conditions:

- Entering CP console function mode
- Loading a disabled PSW
- Loading an enabled PSW with no active I/O in process
- Logging on or off

When the virtual machine becomes executable again, it is put back on the executable list in Q1. If dropped from Q1, the virtual machine is placed directly in Q2 and remains there even though it may exhaust its allotted amount of processor usage. Virtual machines with this option are thus considered for dispatching more frequently than other virtual machines.

Note, however, that these options can impact the response time of interactive users and that only one favored percentage user is allowed at any given time.

PRIORITY

The VM/370 operator can assign specific priority values to different virtual machines. In so doing, the virtual machine with a higher priority is considered for dispatching before a virtual machine with a lower priority. User priorities are set by the following class A command:

```
SET PRIORITY userid nn
```

where userid is the user's identification and nn is an integer value from 1 to 99. The value of nn affects the user's dispatching priority in relation to other users in the system. The priority value (nn) is one of the factors considered in VM/370's dispatching algorithm. Generally, the lower the value of nn, the more favorable the user's position in relation to other users in VM/370's dispatch queues.

RESERVED PAGE FRAMES

VM/370 uses chained lists of available and pageable pages. Pages for users are assigned from the available list, which is replenished from the pageable list.

Pages that are temporarily locked in real storage are not available or pageable. The reserved page function gives a particular virtual machine an essentially "private" set of pages. The pages are not locked; they can be swapped, but only for the specified virtual machine. Paging proceeds using demand paging with a "reference bit" algorithm to select the best page for swapping. The number of reserved page frames for the virtual machine is specified as a maximum. The page selection algorithm selects an available page frame for a reserved user and marks that page frame "reserved" if the maximum specified for the user has not been reached. If an available reserved page frame is encountered for the reserved user selection, it is used whether or not the maximum has been reached.

The maximum number of reserved page frames is specified by a class A command of the following format:

```
SET RESERVE userid xxx
```

where xxx is the maximum number required. If the page selection algorithm cannot locate an available page for other users because they are all reserved, the algorithm forces the use of reserved pages. This function can be specified in only one virtual machine at any one time.

Note: xxx should never approach the total available pages, since CP overhead is substantially increased in this situation, and excessive paging activity is likely to occur in other virtual machines.

VIRTUAL=REAL

For this option, the VM/370 nucleus must be reorganized to provide an area in real storage large enough to contain the entire virtual=real machine. In the virtual machine, each page from page 1 to the end is in its true real storage location; only its page zero is relocated. The virtual machine is still run in dynamic address translation mode, but

since the virtual page address is the same as the real page address, no CCW translation is required. Since CCW translation is not performed, no check is made to ensure that I/O data transfer does not occur into page zero or any page beyond the end of the virtual=real machine's storage.

Systems that are generated with the virtual=real option use the system loader (DMKLD00E). For information about generating a virtual=real system, see the VM/370 Planning and System Generation Guide.

Figure 12 is an example of a real storage layout with the virtual=real option. The V=R area is 128K and real storage is 512K.

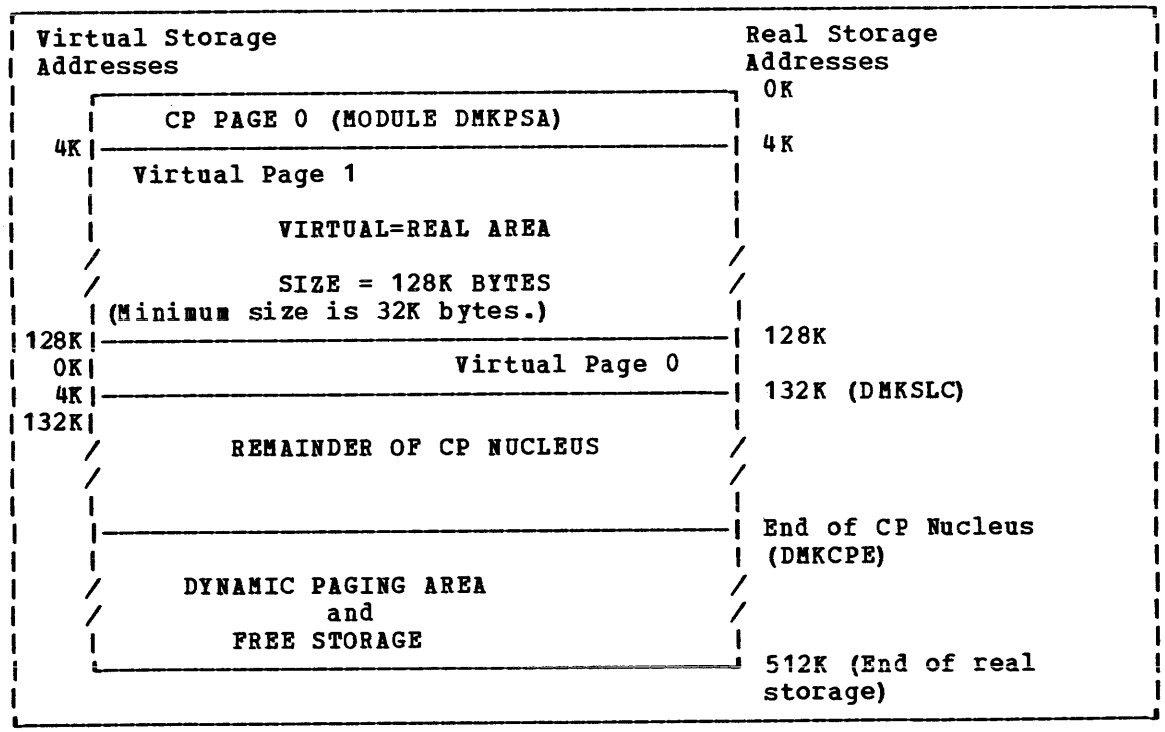


Figure 12. Storage Layout in a Virtual=Real Machine

There are several considerations for the virtual=real option that affect overall system operation:

1. The area of contiguous storage built for the virtual=real machine must be large enough to contain the entire addressing space of the largest virtual=real machine. The virtual=real storage size that a VM/370 system allows is defined during system generation when the option is selected.

2. The storage reserved for the virtual=real machine can only be used by a virtual machine with that option specified in the VM/370 directory. It is not available to other users for paging space, nor for VM/370 usage until released from virtual=real status by a system operator via the CP UNLOCK command. Once released, VM/370 must be loaded again before the virtual=real option can become active again.

3. The virtual machine with the virtual=real option operates in the preallocated storage area with normal CCW translation in effect until the CP SET NOTRANS ON command is issued. At that time, with several exceptions, all subsequent I/O operations are performed from the virtual CCWs in the virtual=real space without translation. The exceptions occur under any of the following conditions:

   - SIO tracing active
   - First CCW not in the V=R region
   - I/O operation is a sense command
   - I/O device is a dial-up terminal
   - I/O is for a nondedicated device
     (spooled unit record console virtual CTCA
     or minidisks that are less than a full volume)
   - Pending device status
   - I/O device has an alternate path

   Any of the above conditions will force CCW translation. Since minidisks are nondedicated devices, they may be used by programs running in the V=R region even though CP SET NOTRANS ON is in effect.

4. If the virtual=real machine performs a virtual reset or IPL, then the normal CCW translation goes into effect until the CP SET NOTRANS ON command is again issued. This permits simulation of an IPL sequence by CP. Only the virtual=real virtual machine can issue the command. A message is issued if normal translation mode is entered.

5. A virtual=real machine is not allowed to IPL a named or shared system. It must IPL by device address.

6. When NOTRANS is in effect for a virtual=real machine, no meaningful SEEK data is collected by MONITOR operations.

AFFINITY

This virtual machine option allows virtual machines that operate on attached processor systems to select, if desired, the processor of their choice for program execution. The selection can be made by the VM/370 directory OPTION statement, or it can be made dynamically by an operand of the CP SET command:

For class G users

```
SET AFFINITY {nn  }
             {OFF}
```

For class A users

```
SET AFFINITY userid {nn }
                   {ON }
                   {OFF}
```

where nn is the processor address of the main or the attached processor.

In application, the affinity setting of a virtual machine implies a preference of operation to either (or neither) processor. Affinity of

operation for a virtual machine means that the program of that virtual machine will be executed on the selected or named processor. It does not imply that supervisory functions and the CP housekeeping functions associated with that virtual machine will be handled by the same processor.

In attached processor systems, all real I/O operations and associated interrupts are handled by the main processor. Virtual I/O initiated on the attached processor that is mapped to real devices must transfer control to the main processor for real I/O execution. Therefore, benefits may be realized in a virtual machine "mix" by relegating those virtual machines that have a high I/O-to-compute ratio to the main processor, and those virtual machines that have a high compute-to-I/O ratio to the attached processor. Such decisions should be carefully weighed as every virtual machine is in contention with other virtual machines for resources of the system.

A more important use of the affinity setting would be in applications where there are virtual machine program requirements for special hardware features that are available on one processor and not the other. Such features could be a performance enhancement such as virtual machine assist (described later in the text) or a special RPQ that is a requirement for a particular program's execution.

VIRTUAL MACHINE ASSIST FEATURE

The virtual machine assist feature is a processor hardware feature. It improves the performance of VM/370. Virtual storage operating systems, which run in problem state under the control of VM/370, use many privileged instructions and SVCs that cause interrupts that VM/370 must handle. When the virtual machine assist feature is used, many of these interrupts are intercepted and handled by the processor; and, consequently, VM/370 performance is improved. See VM/370 Planning and System Generation Guide for a list of the processors on which virtual machine assist is available.

The virtual machine assist feature intercepts and handles interruptions caused by SVCs (other than SVC 76), invalid page conditions, and several privileged instructions. An SVC 76 is never handled by the assist feature; it is always handled by CP. The processing of the following privileged instructions is handled by this feature:

LRA	(load real address)
STCTL	(store control)
RRB	(reset reference bit)
ISK	(insert storage key)
SSK	(set storage key)
IPK	(insert PSW key)
STNSM	(store then AND system mask)
STOSM	(store then OR system mask)
SSM	(set system mask)
LPSW	(load PSW)
SPKA	(set PSW key from address)

Although the assist feature was designed to improve the performance of VM/370, virtual machines may see a performance improvement because more resources are available for virtual machine users.

Using the Virtual Machine Assist Feature

Whenever you IPL VM/370 on a processor with the virtual machine assist feature, the feature is available for all VM/370 virtual machines. However, the system operator's SET command can make the feature unavailable to VM/370 and, subsequently, available again for all users. The format of the system operator's SET command is:

```
SET SASSIST {ON } [[PROC] xx]
            {OFF}
```

If you do not know whether or not the virtual machine assist feature is available to VM/370, use the class A and E QUERY command. For a complete description of the Class A and E QUERY and SET commands, see the VM/370 Operator's Guide.

If the virtual machine assist feature is available to VM/370 when you log on your virtual machine, it is also supported for your virtual machine. If your VM/370 directory entry has the SVCOFF option, the SVC handling portion of the assist feature is not available when you log on. The class G SET command can disable the assist feature (or only disable SVC handling). It can also enable the assist feature, or if the assist feature is available, enable the SVC handling. The format of the command is:

```
SET ASSIST {[ON] [SVC  ] [TMR  ]}
           {     [NOSVC] [NOTMR]}
           {OFF                  }
```

You can use the class G QUERY SET command line to find whether you have full, partial, or none of the assist feature available. For a complete description of the Class G QUERY and SET commands, see the VM/370 CP Command Reference for General Users.

Restricted Use of the Virtual Machine Assist Feature

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

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

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

Whenever a virtual machine issues a TRACE command with the SVC, PRIV, BRANCH, INSTRUCT, or ALL operands, the virtual assist feature is automatically turned off for that virtual machine. The assist feature is turned on again when the tracing is completed. If the QUERY SET command line is issued while SVCs or program interrupts are being traced, the response will indicate the assist feature is off.

VM/370 EXTENDED CONTROL-PROGRAM SUPPORT (ECPS)

VM/370 Extended Control-Program Support (ECPS) improves the performance of the processor when executing VM/370 beyond the improvement attained by the virtual machine assist feature described above. See VM/370 Planning and System Generation Guide for a list of the processors on which ECPS is available. ECPS consists of three parts: CP assist, expanded virtual machine assist, and virtual interval timer assist. A detailed description of ECPS is provided in "Appendix A. VM/370 Extended Control-Program Support."

CP Assist

The CP assist part of ECPS assists various routines that are frequently used by VM/370. Because these routines are assisted by the hardware without involving VM/370, performance of the VM/370 system is improved. The high-use paths of the following functions are assisted:

- Get Free Space (DMKFRE)
- Release Free Space (DMKFRE)
- Untranslate CSW (DMKUNT)
- Free CCW Storage (DMKUNT)
- Locate Virtual I/O Control Block (DMKSCN)
- Locate Real I/O Control Block (DMKSCN)
- Lock a page (DMKPTR)
- Unlock a page (DMKPTR)
- Common CCW command processing (DMKCCW)
- Decode First CCW command (DMKCCW)
- Decode following CCW command (DMKCCW)
- TRANBRNG subroutine (DMKCCW)
- TRANLOCK subroutine (DMKCCW)
- Invalidate page table subroutine (DMKVAT)
- Invalidate segment table subroutine (DMKVAT)
- Main entry to dispatch (DMKDSP)
- Dispatch a block or virtual machine (DMKDSP)
- SVC 8 (LINK)
- SVC 12 (RETURN)
- Locate changed shared pages (DMKVMA)

Expanded Virtual Machine Assist

Expanded virtual machine assist extends the level of handling of the following privileged instructions:

- LPSW
- STNSM
- STOSM
- SSM

In addition, expanded virtual machine assist handles the processing of the following privileged instructions not handled by the virtual machine assist feature:

- PTLB
- SIO
- SPT
- SCKC
- STPT
- TCH

Virtual Interval Timer Assist

Virtual interval timer assist provides hardware updating of the virtual interval timer at virtual location X'50'. This results in an update frequency of approximately 300 times per second, the same as for the real interval timer. Procedures that use the virtual interval timer for job accounting, performance measurements, and the like, will therefore generate more accurate and repeatable time data than they would if the virtual timer was being updated by CP routines.

Using the VM/370 Extended Control-Program Support

VM/370 Extended Control-Program Support (ECPS) is controlled at two levels: the VM/370 system and the virtual machine.

At the VM/370 system level, ECPS is automatically enabled when the system is loaded. The class A command:

```
set cpassist off
```

will disable both CP assist and expanded virtual machine assist. The class A command:

```
set sassist off
```

disables only the expanded virtual machine assist part of ECPS as well as the virtual machine assist. CP assist is the only part of ECPS that is truly independent.

At the virtual machine level, whenever ECPS is enabled on the system, both expanded virtual machine assist and virtual interval timer assist are automatically enabled when you log on. If you issue the class G command:

```
set assist off
```

both assists as well as the existing virtual machine assist are disabled. If you issue:

```
set assist notmr
```

only the virtual interval timer assist is disabled. If CP assist is disabled for the system, the class A command:

```
set sassist on
```

will enable the virtual machine assist. You can then enable virtual machine assist and virtual interval timer assist for your virtual machine by issuing the class A command:

```
set assist on tmr
```

Restricted Use of ECPS

The restrictions on the use of ECPS are the same as those described for the virtual machine assist feature with one addition. Whenever a virtual machine traces external interrupts, the virtual interval timer

assist is automatically disabled. When external interrupt tracing is completed, virtual interval timer assist is reenabled.

Virtual Machine Communication Facility

The Virtual Machine Communication Facility (VMCF) allows any logged-on user of VM/370 to transfer messages, control data, data files, or combinations of all three to another virtual machine running under the same VM/370 system. Information is transferred directly from one virtual storage to the other virtual storage with CP buffering the information. Only one data page frame must be locked at any one time. The amount of data that can be transferred is limited only by the virtual storage sizes of the virtual machines involved.

VMCF contains five data movement and seven control functions and is invoked by a virtual machine via the DIAGNOSE interface (code X'0068'). A special external interrupt code, X'4001', notifies a virtual machine that a VMCF communication is pending. A virtual machine can have a maximum of 50 messages active at any one time. The number of messages is an equate in the DMKVMC module and can be changed to accommodate different VM/370 storage sizes.

VMCF Diagnose Interface

When a virtual machine issues a DIAGNOSE instruction with a function code of X'0068', the rx register contains the virtual address, doubleword-aligned, of a 40-byte parameter list. This parameter list (VMCPARM) contains a hexadecimal code to identify the specific VMCF subfunction. It also contains the data addresses, data lengths, and control information that are required to execute the particular subfunction.

The DIAGNOSE instruction, a privileged operation, is processed by DMKPRV which passes control to DMKHVC, the DIAGNOSE interface module. DMKHVC, in turn, validates the function code and, if the code is X'0068', turns control over to DMKVMC, the VMCF module. DMKVMC validates the VMCPARM address and length, the subfunction code, and passes control to the appropriate subroutine. The VMCF subfunctions and their codes are as follows:

Code	Subfunction
X'0000'	Allow virtual machine communication
X'0001'	Disallow virtual machine communication
X'0002'	Initiate a SEND request
X'0003'	Initiate a SEND/RECV request
X'0004'	Initiate a SENDX request
X'0005'	Accept data from a SEND or SEND/RECV request
X'0006'	Cancel specific request you initiated
X'0007'	Reply to a SEND/RECV request
X'0008'	Reject further incoming communications
X'0009'	Resume accepting communications
X'000A'	Notify a user that you are ready for communications
X'000B'	Reject a specific incoming communication

Special VMCF External Interrupt

Whenever a source virtual machine uses VMCF to correspond with another virtual machine (sink), the sink is notified of the pending communication via a special external interrupt (code X'4001'). When this interrupt is unstacked and processed, a copy of the information in the source's parameter list is passed to the sink in an external interrupt buffer. The buffer is defined when a user allows virtual machine communication. The contents are referred to as the external interrupt message header.

When certain transactions (SEND, SEND/RECV, SENDX) have been completed, a final response external interrupt is passed back to the source. The message header associated with this interrupt contains residual counts pertaining to the transferred data and data transfer return codes.

VMCF Control Blocks and Data Areas

Figure 13 shows the relationship between the various VMCF control blocks and data areas. When a virtual machine allows virtual machine communication, VMCF generates a master VMCBLOK and places it at the head of a queue pointed to by the VMCPNT field of the user's VMBLOK. Two fields in this master VMCBLOK define the address (VMCVADA) and length (VMCLENA) of the user's external interrupt buffer. The length must include the maximum size of any potential SENDX data in addition to the 40 bytes for the external interrupt message header.

When a source virtual machine executes a VMCF subfunction, a VMCBLOK is built, initialized with data from the parameter list (VMCPARM), and stacked on the VMCBLOK queue pointed to by the VMCPNT field in the sink's VMBLOK. If an XINTBLOK for a X'4001' external interrupt has not already been stacked for the sink machine, DMKVMC builds one and stacks it on the XINTBLOK queue pointed to by the VMPXINT field in the sink's VMBLOK. VMCF external interrupts are assigned a sort code of X'7FFFFFFF', giving them the lowest priority in the external interrupt queue. Each virtual machine clears its own VMCF control blocks.

Special Messages Facility

The Special Message Facility allows users to send special messages to a virtual machine via the SMSG command. In the Special Message environment, CP acts as a source machine with the receiver of special messages being the sink. This relieves the burden from the issuer of SMSG of having to perform authorization and other setup necessary for sending messages to the receiving virtual machine. This is performed by CP.

The issuer of SMSG is responsible for sending message text that is meaningful to the receiving virtual machine. The format and handling of special messages is entirely up to the receiving machine, which may be one designed by the installation or prepared by others.

Before the receiving virtual machine can accept special messages, it must be running with the Special Message flag ON, and it must have issued AUTHORIZE (via DIAGNOSE X'68') with CP. The authorization includes supplying the External Interrupt Buffer address and size. To ensure receiving the entire message, the receiving virtual machine

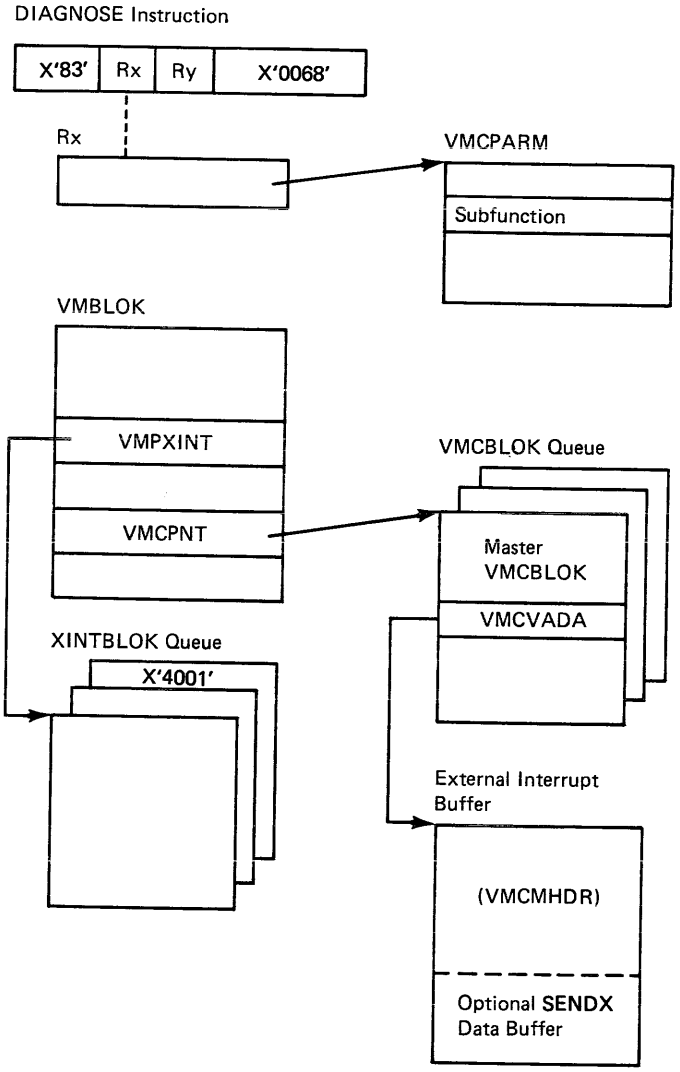


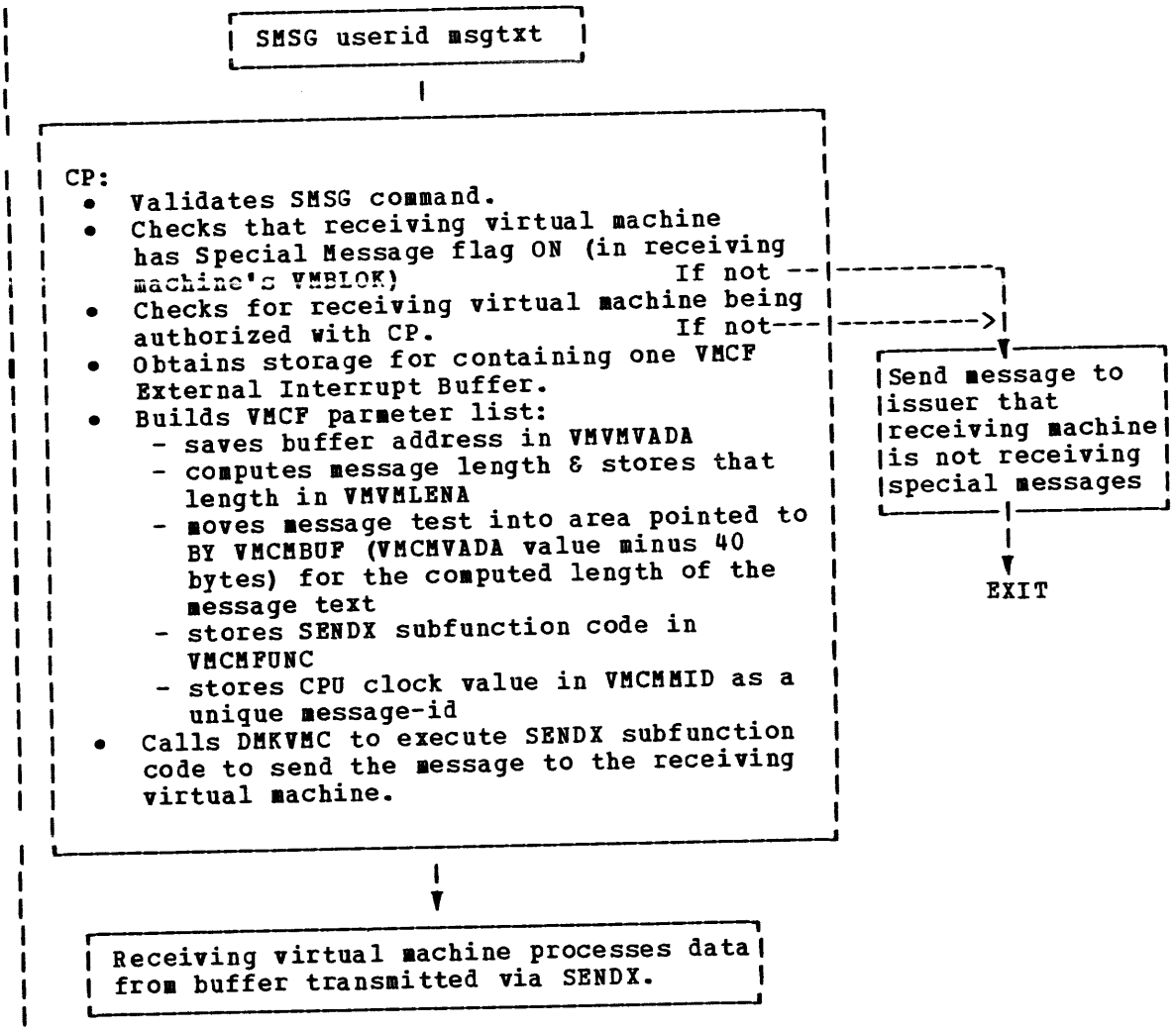
Figure 13. VMCF Control Block Relationships

| should specify the size as 169 bytes (room for a 40-byte header and a
| 120- to 129-byte message buffer, depending on the length of the command
| and userid).

| Setting SMSG ON can be accomplished by setting the SMSG flag on in
| the VMCF parameter list when issuing an AUTHORIZE. It may also issue
| the CP command SET SMSG ON. Either method sets the Special Message flag
| on in the VMBLOK. When this is done, any other virtual machine can
| issue the SMSG command to the userid of the receiving virtual machine.

If the receiving virtual machine chooses not to accept special messages at any time, it can merely issue SET SMSG OFF. CP would then inform any machine issuing the SMSG command that the virtual machine is not receiving special messages. When it is ready to resume accepting special messages, the virtual machine need only to issue SET SMSG ON.

The following shows the processing when an SMSG command is issued.



During a Special Message session, the following error messages could be sent back to the issuer of the SMSG command:

```
DMKMSG003E:  INVALID OPTION - option
DMKMSG020E:  USERID MISSING OR INVALID
DMKMSG045W:  userid NOT LOGGED ON
DMKMSG057E:  userid NOT RECEIVING; [DISCONNECTED | MSG OFF |
               SMSG OFF | NOT AUTHORIZED | WNG OFF]
```

VM/VS Handshaking

The VM/VS Handshaking feature provides a communication path between CP and virtual machine operating systems that makes each system control program aware of certain capabilities or requirements of the other.

The following is a discussion of VM/VS Handshaking as it relates to OS/VS1. Functions of VM/VS Handshaking incorporated in the VM/370 control program are available and applicable to any operating system that can be system generated to use this VM/370 enhancement.

VM/VS Handshaking for OS/VS1 performs the following functions:

- Closes CP spool files when the VS1 job output from its DSO, terminator, and output writer is complete
- Processes VS1 pseudo page faults
- Provides an optional nonpaging mode for VS1 when it is run in the VM/370 environment

When a VS1 virtual machine with the handshaking feature is loaded (via IPL), its initialization routines determine whether the handshaking feature should be enabled. First, VS1 determines if it is running under the control of VM/370 by issuing a STIDP (Store Processor ID) instruction. STIDP returns a version code; a version code of X'FF' indicates VS1 is running with VM/370. If VS1 finds a version code of X'FF', it then issues a DIAGNOSE (X'00') instruction to store the VM/370 extended-identification code. If an extended-identification code is returned to VS1, VS1 knows that VM/370 supports handshaking; if nothing is returned to VS1, VM/370 does not support handshaking. At this time or any time after IPL, the operator of the VS1 virtual machine can issue the CP SET PAGEX ON command to enable the pseudo page fault handling portion of handshaking. If the VS1 virtual machine is in the nonpaging mode and, if the pseudo page fault handling is active, full handshaking support is available.

Because the VS1 system does no paging, any ISAM programs run under VS1 are treated by VM/370 as though they are running in an ADDRSPC=REAL partition. Therefore, the ISAM option is required for the VS1 machine to successfully execute the ISAM program.

Closing CP Spool Files

If the handshaking feature is active, VS1 closes the CP spool files when its job output from the DSO, terminator, and output writer is complete. Once the spool files are closed, VM/370 processes them and they are sent to the real printer or punch. During its job termination processing, VS1 issues a DIAGNOSE (X'08') instruction to pass the CP CLOSE command to VM/370 for each CP spool file.

Pseudo Page Faults

A page fault is a program interruption that occurs when a page marked "not in storage" is referred to by an instruction with an active page. The virtual machine referring to the page is placed in a wait state while the page is brought into real storage. Without the handshaking feature, the entire VS1 virtual machine is placed in page wait by VM/370 until the needed page is available.

However, with the handshaking feature, a multiprogramming (or multitasking) VS1 virtual machine can dispatch one task while waiting for a page request to be answered for another task. VM/370 passes a pseudo page fault (program interrupt X'14') to VS1. When VS1 recognizes the pseudo page fault, it places only the task waiting for the page in page wait and can dispatch another tasks.

When a page fault occurs for a VS1 virtual machine, VM/370 checks that the pseudo page fault portion of handshaking is active and that the VS1 virtual machine is in EC mode and enabled for I/O interruptions. Then, VM/370 reflects the page fault to VS1 by:

- Storing the virtual machine address that caused the page fault at location X'90' (the translation exception address)
- Indicating a program interruption (interrupt code X'14') to VS1
- Removing the VS1 virtual machine from page wait and execution wait

When VS1 recognizes program interruption code X'14', it places the associated task in wait state. VS1 can then dispatch other tasks.

When the requested page becomes available in real storage, VM/370 indicates the same program interruption to VS1, except that the leftmost bit in the translation exception address field is set on to indicate completion. VS1 removes the task from page wait; the task is then eligible to be dispatched.

VS1 Nonpaging Mode

When VS1 runs under the control of VM/370, it executes in nonpaging mode if:

- Its virtual storage size is equal to the size of the VM/370 virtual machine
- Its virtual machine size is at least 1024K bytes and no more than 4096K bytes. For VS1 Release 6, the maximum size is 16,370K bytes.
- The VM/VS Handshaking feature is available.

When VS1 executes in nonpaging mode, it uses fewer privileged instructions and avoids duplicate paging. The VS1 Nucleus Initialization Program (NIP) fixes all VS1 pages to avoid the duplicate paging.

Note: The working set size may be larger for a VS1 virtual machine in nonpaging mode than for one in paging mode.

Miscellaneous Enhancements

A VS1 virtual machine with the handshaking feature avoids many of the instructions or procedures that would duplicate the function that VM/370 provides. For example, VS1 avoids:

- ISK (Insert Storage Key) instructions and uses a key table
- Seek separation for 2314 direct access devices
- ENABLE/DISABLE sequences in the VS1 I/O Supervisor (IOS)
- TCH (Test Channel) instructions preceding SIO (Start I/O) instructions

CP Interruption Handling

Interruption processing occurs within the CP environment. More than 30 modules control the process of interrupting events brought about by CP or virtual machine activity. Each module handles a particular I/O device or class or a function of CP, (for example: timers, paging, SVCs). For an overview of interruption handling, see Figure 14.

Program Interruption

Program interruptions occur in two states. If the CPU is in the supervisor state, the interruption indicates a system failure in the CP nucleus and causes a system abnormal termination. If the CPU is in the problem state, a virtual machine is in execution. If the program interruption indicates that the Dynamic Address Translation (DAT) feature has an exception, a virtual machine issued a privileged instruction, or a protection exception occurred for a shared segment system, CP takes control and performs any required processing to satisfy the exception. Usually, the interruption is not apparent to the virtual machine. Most other program interruptions result from virtual machine processing and are reflected to the virtual machine for handling.

When a program interruption occurs, the program interruption handler (DMKPRG) is entered. Program interruptions can result from:

- Normal paging requests
- A paging request by a virtual machine in EC mode (virtual relocate mode)
- Privileged instructions
- Program errors

For information about paging requests, see "Allocation Management" in this section.

Privileged Instructions

If a program interruption is caused by the virtual machine issuing a privileged instruction when it is running in supervisor state, DMKPRVLG obtains the address of the privileged instruction and determines the type of operation requested. If the virtual machine was running in problem state, the interruption is reflected back to the virtual machine.

I/O PRIVILEGED INSTRUCTIONS

DMKPRVLG transfers control to the virtual I/O executive program (DMKVSIEX).

TYPE	MODULE
SVC	DMKSVCIN
External	DMKPSAEX
Machine Check	DMKMCHIN
I/O	DMKIOSIN
Program Check	DMKPRGIN

Interrupt Handler Modules

Interrupt From	Action/Module
Unknown channel	Ignored - DMKDSPCH
Unsolicited device end	Build IOBLOK
and for:	
Console	DMKCNSIN
3270s on bisync lines	DMKRGA or DMKRGB
Local 3270, 3158, and 3066 consoles	DMKGRF
Unit record, real spooling	DMKRSPEX
Solicited device end	DMKSTKIO
Channel error	DMKCCHNT
Monitor tape I/O operation	DMKMONIO
Dedicated device error - DASD	DMKDASER
Dedicated device error - Tape	DMKTAPER
3270 bisync line and channel errors	DMKBSG
Recoverable	DMKSTKIO
Unrecoverable	DMKIOERR

I/O Interrupt Handler (DMKIOS) Actions

Reason for Program Check	Module
Normal paging	DMKPTRAN
Paging - virtual machine in EC mode	DMKVAT
Supervisor State	DMKDMP
Privileged instruction	DMKPRVLG
DIAGNOSE	DMKHVC
Timers	DMKTMR
Virtual Machine I/O	DMKVSIEX
console	DMKVCNEX
unit record, virtual spooling	DMKVSPEX

Program Check Interrupt Handler (DMKPRG) Actions

Figure 14. Overview of Interruption Handling

NON-I/O PRIVILEGED INSTRUCTIONS

DMKPRVLG simulates valid non-I/O privileged instructions and returns control to DMKDSPCH. For invalid non-I/O privileged instructions, the routine sets an invalid interruption code and reflects the interruption to the virtual machine. For the privileged instructions (SCK, SCKC, STCKC, SPT, and STPT) that affect the TOD clock, CPU timer, and TOD clock comparator, control is transferred to DMKTMR by DMKPRVLG. Other instructions that are simulated are LPSW, SSM, SSK, ISK, IPTE, and DIAGNOSE.

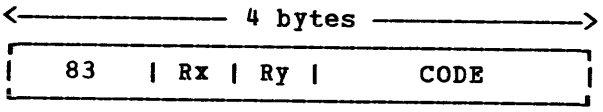
Although the CS and CDS instructions are nonprivileged, they are not part of the standard instruction set on IBM System/370 Models 135, 135-3, 138, 145, 145-3, and 148; VM/370 simulates these instructions on these models that do not have the optional hardware feature installed.

System/370 EC mode non-I/O privileged instruction simulation includes the following:

Code	Definition
SCK	Set Clock
SCKC	Set Clock Comparator
STCKC	Store Clock Comparator
SPT	Set CPU Timer
STPT	Store CPU Timer
STNSM	Store and AND System Mask
STOSM	Store and OR System Mask
STIDP	Store CPU Identification
STIDC	Store Channel Identification
LCTL	Load Control
STCTL	Store Control
LRA	Load Real Address
RRB	Reset Reference Bit
PTLB	Purge Table Look-aside Buffer
IPK	Insert PSW Key
SPKA	Set PSW Key From Address

DIAGNOSE INSTRUCTION IN A VIRTUAL MACHINE

The DIAGNOSE instruction cannot be used in a virtual machine for its normal function. If a virtual machine attempts to execute a DIAGNOSE instruction, a program interrupt returns control to CP. Since a DIAGNOSE instruction issued in a virtual machine results only in returning control to CP and not in performing normal DIAGNOSE functions, the instruction is used for communication between a virtual machine and CP. The machine language format of DIAGNOSE is:



where:

83 is X'83' and interpreted by the assembler as the DIAGNOSE instruction.

Note: There is no mnemonic for DIAGNOSE.

Rx specifies a register containing the address of the VMCPARM parameter list.

Ry is a register that contains a return code.

CODE is X'68' and specifies that you are requesting execution of a VMCF.

The operand storage addresses, passed to the DIAGNOSE interface in Rx and Ry, must be real addresses to the virtual machine issuing the DIAGNOSE.

The code is a two-byte hexadecimal value that CP uses to determine what function to perform. The codes defined for the general VM/370 user are described in this section. The code must be a multiple of 4. Codes X'00' through X'FC' are reserved for IBM use, and codes X'100' through X'1FC' are reserved for users.

Because DIAGNOSE operates differently in a virtual machine than it does in a real machine, a program should determine that it is operating in a virtual machine before issuing a diagnose instruction, and prevent execution of a DIAGNOSE when in a real machine. The Store Processor ID (STIDP) instruction provides a program with information about the processor in which it is executing, including the processor version number. If STIDP is issued from a virtual machine, the version number will be X'FF' in the first byte of the CPUID field.

A virtual machine issuing a diagnose instruction should run with interrupts disabled. This prevents loss of status information pertaining to the diagnose operation such as condition codes and sense data.

DIAGNOSE Code X'00' -- Store Extended-Identification Code

Execution of DIAGNOSE code X'00' allows a virtual machine to examine the VM/370 extended-identification code. For example, an OS/VS1 virtual machine issues a DIAGNOSE code X'00' instruction to determine if the version of VM/370 under which it is executing supports the VM/VS Handshaking feature. If the extended-identification code is returned to VS1, VM/370 supports handshaking; otherwise, it does not.

The register specified as Rx contains the doubleword aligned virtual storage address where the VM/370 extended-identification code is to be stored. The Ry register contains the number of bytes to be stored entered as an unsigned binary number.

If the VM/370 system currently executing does not support the DIAGNOSE code X'00' instruction, no data is returned to the virtual machine. If it does support the DIAGNOSE code X'00' instruction, the following data is returned to the virtual machine (at the location specified by Rx):

Field	Description	Characteristics
System Name	"VM/370"	8 bytes, EBCDIC
Version Number	The first byte is the version number, the second byte is the level, and the third byte is the PLC (Program Level Change) number.	3 bytes, hexadecimal
Version Code	VM/370 executes the STIDP (Store Processor ID) instruction to determine the version code.	1 byte, hexadecimal
MCEL	VM/370 executes the STIDP instruction to determine the maximum length of the MCEL (Machine Check Extended Logout) area.	2 bytes, hexadecimal
Processor Address	VM/370 executes the STAP (Store Processor Address) instruction to determine the processor address.	2 bytes, hexadecimal
Userid	The userid of the virtual machine issuing the DIAGNOSE.	8 bytes, EBCDIC
Program Product Bit Map	Reserved for IBM use	8 bytes, hexadecimal

If VM/370 is executing in a virtual machine, another 24 bytes, or less, of extended identification data is appended to the first 24 bytes described above. Up to five nested levels of VM/370 virtual machines are supported by this diagnose instruction resulting in a maximum of 120 bytes of data that can be returned to the virtual machine that initially issued the diagnose instruction.

Upon return, Ry contains its original value less the number of bytes that were stored.

No completion code is returned, and the condition code remains unchanged.

DIAGNOSE Code X'04' -- Examine Real Storage

Execution of a DIAGNOSE code X'04' allows a user with command privilege class C or E to examine real storage. The register specified as Rx contains the virtual address of a list of CP (real) addresses to be examined. The Ry register contains the count of entries in the list. Ry+1 contains the virtual address of the result field. The result field contains the values retrieved from the specified real locations.

Note: The request and result tables must be in the same page of virtual storage, and that page must be resident in real storage, at the time the DIAGNOSE is executed. This is guaranteed if the instruction itself is also in the same page.

In the attached processor environment, each processor has a prefix register to relocate addresses between 0 and 4095 to another page frame in main storage. The prefix register enables each processor to use a different page frame in order to avoid conflict with the other processor for such activity as interrupt code recording. Thus, the range 0 through 4095 refers to different areas of storage, depending upon which processor generates the address.

All references to main storage from either processor are handled as if they were made on the main processor. Existing user programs remain valid for performance data; in the attached processor environment, they receive the statistics for the main processor.

References to the PSA of the attached processor may be made as follows: first, retrieve the value of PREFIXB, the value of the prefix register for the other processor (the attached processor in this case). Next, specify addresses that are the sum of the value of PREFIXB and the PSA displacement. References to 0 through 4095 are made by summing the value of PREFIXA and the PSA displacement to form the request address. Several system values that are processor independent are maintained in 0 through 4095, such as the restart PSW and the trace table vectors.

DIAGNOSE Code X'08' -- Virtual Console Function

The execution of DIAGNOSE code X'08' allows a program executing in supervisor mode in a virtual machine to perform a CP command. The register specified as Rx contains the address, in virtual storage, of the data area defining the CP command and parameters. The Ry register contains flags in the high-order byte, and contains the command length (up to 132 characters) in the three low-order bytes. If the first bit of the high-order bytes is on (X'80'), it indicates that the virtual machine issuing the DIAGNOSE code X'08' instruction wants CP to reject passwords from the terminal for AUTOLOG and LINK commands. This will be done only if the installation has elected to use the password suppression facility (PSUPRS=YES parameter of the SYSJRL macro in DMKSYS).

If the second bit of the high-order byte is on (X'40'), it indicates that the virtual machine issuing the CP command requires the response to that command to be returned to the virtual storage area specified by register Rx+1. The length of this area must be passed in register Ry+1. The following example illustrates how DIAGNOSE code X'08' would be issued to perform the CP command, QUERY, to determine the number of input and output spool files:

```
LA       6,CMMD
LA      10,CMMDL
DC      X'83',X'6A',XL2'0008'
.
.
.
CMMD    DC       C'QUERY FILES'
CMMDL   EQU      *-CMMD
.
.
.
```

If the high-order byte of the Ry register is an X'40', the output of the command is returned to the user's virtual storage area; otherwise, it is at the user's terminal. A completion code is returned to the user as a value in the register specified as Ry. In the example above, it would be register 10. A completion code of 0 signifies normal completion. If there is an error, the completion code is the binary value of the numeric portion of the error message. For instance, the error message

DMKCFM045E userid NOT LOGGED ON

returns "045" in the Ry register. The condition code remains unchanged.

If a CP command is to be executed, the instruction acts as a no-op. If Ry contains a zero, CP is entered. The BEGIN command then returns control to your program.

The user can have the response returned in a buffer rather than to his console. He is entirely responsible for setting up the buffer, providing the buffer address, and examining and processing the returned response (contents of the buffer). To have the response returned in the buffer, the user sets up registers as follows:

- Rx contains the virtual address of the CP console functions command and parameters.
- Rx+1 contains the virtual address of the buffer to receive the response.
- Ry contains the length of the CP console function command (up to 132 characters) with an X'40' in the high-order byte.
- Ry+1 contains the length of the response buffer (a positive number not greater than 8192).

Neither Rx nor Ry can be register 15; Rx and Ry cannot be consecutive registers.

When returned in a buffer, the DIAGNOSE code X'08' output will also have the following successful or unsuccessful conditions set:

- If the response fits into the user's buffer, the condition code is set to zero and the number of response characters returned in the user's buffer is returned in Ry+1.
- If the response does not fit in the user's buffer, the condition code is set to one, and the amount of overflow (number of response bytes that would not fit in the user's buffer) is returned in register Ry+1.

DIAGNOSE Code X'0C' -- Pseudo Timer

Execution of DIAGNOSE code X'0C' causes CP to store four doublewords of time information in the user's virtual storage. The register specified as Rx contains the address of the 32-byte area where the time information is to be stored. The address must be on a doubleword boundary.

The first eight bytes contain the month/day-of-month/year. The next eight bytes contain the time of day in hours:minutes:seconds. One-hundredths of seconds are not returned. The last 16 bytes contain the virtual and total processor time used by the virtual machine that issued the DIAGNOSE. These times are expressed as doubleword, unsigned integers, in microseconds. No completion code is returned, and the condition code remains unchanged.

DIAGNOSE Code X'10' -- Release Pages

Pages of virtual storage can be released by issuing a DIAGNOSE code X'10'. When a page is released, it is considered all zero. The register specified by Rx contains the address of the first page to be released, and the Ry register contains the address of the last page to be released. Both addresses must be on page boundaries. A page boundary is a storage address whose low-order three digits, expressed in hexadecimal, are zero. No completion code is returned, and the condition code remains unchanged.

| Note: DIAGNOSE code X'10' is not to be used to release discontiguous
| storage. See DIAGNOSE code X'64' for releasing discontiguous storage.

DIAGNOSE Code X'14' -- Input Spool File Manipulation

Execution of DIAGNOSE code X'14' causes DMKDRDER to perform input spool file manipulation. Depending upon the value of the function subcode, the register specified as Rx contains a buffer address, a copy count, or a spool file identifier. The Ry register, which must be an even register, contains either the virtual address of a spool input card reader or, if Ry+1 contains X'0FFF', a spool file ID number. Ry+1 contains a hexadecimal code indicating the file manipulation to be performed. The codes are:

Code	Function
0000	Read next spool buffer (data record)
0004	Read next print spool file block (SFBLOK)
0008	Read next punch spool file block (SFBLOK)
000C	Select a file for processing
0010	Repeat active file nn times
0014	Restart active file at beginning
0018	Backspace one record
001C	Read next monitor spool file block
0020	Read next monitor spool record
0FFF	Retrieve subsequent file descriptor

On return Ry+1 may contain error codes that further define a returned condition code of 3.

Condition Code	Ry+1	Error
0		Data transfer successful
1		End of file
2		File not found
3	4	Device address invalid
3	8	Device type invalid
3	12	Device busy
3	16	Fatal paging I/O error
3	20	Page already locked for I/O

Subcode X'0000'

Rx = start address of fullpage virtual buffer
Ry = virtual spool reader address

The specified device is checked for a file already activated via DIAGNOSE and, if there is one, the next fullpage buffer is made available to the virtual machine via a call to DMKRPAGT. If no file is active via DIAGNOSE, the chain of reader files is searched for a file for the calling user and connected to the virtual device for further reading. If no file is found, virtual condition code 2 is set. When the end of an active file is reached, the device status settings are tested for "spool continuous."' If not set, virtual condition code 1 is set, indicating end of file. If the device is set for continuous input, the active file is examined to determine whether or not it is a multiple-copy file. If it is, reading is restarted at the beginning of the file. If it is not, the file is closed via DMKVSPCR and the reader chain is searched for another input file. If no other file is found, virtual condition code 1 is set. A specific DIAGNOSE X'14' Subcode X'0000' must be issued to get the first spooled page again.

Subcode X'0004'

Rx = virtual address of a 12-doubleword buffer
Ry = virtual spool reader address

If the specified device is in use via diagnose, the VSPLCTL block is checked to see whether or not this is a repeated call for printer SFBLOKs. If it is, then the chain search continues from the point where the last SFBLOK was given to the virtual machine. In this case, cc = 1 is set when there are no more print files. If this is the first call for an SFBLOK, or if there have been intervening calls for file reading, the spool input chain is searched from the beginning, and cc=2 is set if no files are found.

Note: The virtual buffer specified via Rx must not cross a page boundary or a specification exception will result.

Subcode X'0008'

Rx = virtual address of a 12-doubleword buffer
Ry = virtual spool reader address

Processing for subcode X'0008' is the same as for subcode X'0004', except that only card-image input files are processed.

Note: For both subcode X'0004' and subcode X'0008', the format definition for a VM/370 SFBLOK can be found in the system macro library.

Subcode X'000C'

Rx = file identifier of requested file
Ry = virtual spool reader address

The spool input chain is searched for the file specified. If it is not found, cc=2 is set. If it is found, the file is moved to the head of the chain so that it will be the next file processed by any of the other functions.

Subcode X'0010'

Rx = new copy count for the active file
Ry = virtual spool reader address

The specified device is checked for an active file. If no file is active, cc=2 is set. Otherwise, the copy COUNT for the file is set to the specified value, with a maximum of 255. If the specified count is not positive, a specification exception is generated. If the count is greater than 255, it is adjusted to module 256.

Subcode X'0014'

Rx = start address of virtual fullpage buffer
Ry = virtual spool reader address

The specified device is checked for an active file. If no active file is found, cc=2 is set. Otherwise, the VSPLCTL pointers are reset to the beginning of the file.

Subcode X'0018'

Rx = start address of virtual fullpage buffer
Ry = virtual spool reader address

The specified device is checked for an active file. If no active file is found, cc=2 is set. Otherwise, the file is backspaced one record and the record is given to the user as in subcode X'0000'. If the file is already positioned at the first record, the first record is given to the user.

Subcode X'001C'

Rx = virtual address of a 12-doubleword buffer
Ry = virtual spool reader address

Processing is the same as Subcode X'0008', except that only monitor spool files, as identified by the SFBMON flag is SFBFLAG2, can be handled.

Subcode X'0020'

Rx = start address of fullpage virtual buffer
Ry = virtual spool reader address

Processing is the same as Subcode X'0000', except that only monitor spool files, as identified by the SFBMON flag in SFBFLAG2, can be handled.

Subcode X'0FFF'

Rx = virtual address of a 252-byte buffer
Ry = spool file ID number

If Ry is nonzero, the spool input chain is searched for a file with a matching ID number: If none is found or if one is found that is owned by a different virtual machine, cc=2 is set. The chain search is continued from the file that was found, or from the anchor if Ry is zero, for the next file owned by the caller, independent of file type, class, INUSE flag, etc. If none is found, cc=1 is set. Otherwise, the SFBLOK and the first record of the file (generally, the TAG) are copied to the caller's virtual storage buffer.

DIAGNOSE Code X'18' -- Standard DASD I/O

Input/output operations to a direct access device, of the type used by CMS, can be performed from a virtual machine using DIAGNOSE code X'18'. No I/O interrupts are returned by CP to the virtual machine; the DIAGNOSE instruction is completed only when the READ or WRITE commands associated with the DIAGNOSE are completed. The Rx register contains the virtual device address of the direct access device. The Ry register contains the address of a chain of CCWs. The CCW chain must be in a standard format that CP expects when DIAGNOSE code X'18' is used, as shown below. Register 15 must be loaded by the user with the number of READs or WRITEs in the CCW chain.

A typical CCW string to read or write two 800-byte records is as follows:

```
     SEEK,A,CC,6
     SET SECTOR (not used for 2314/2319)
     SRCH,A+2,CC,5
     TIC,*-8,0,0
     RD or WRT,DATA,CC+SILI,800
     SEEK HEAD,B,CC,6 (omitted if HEAD number unchanged)
     SET SECTOR
     SRCH,B+2,CC,5
     TIC,*-8,0,0
     RD or WRT,DATA+800,SILI,800

A    SEEK and SRCH arguments for first RD/WRT
B    SEEK and SRCH arguments for second RD/WRT
```

The condition codes and completion codes returned are as follows:

cc=0 I/O complete with no errors

cc=1 Error condition. Register 15 contains one of the following:

- R15=1 Device not attached
- R15=2 Device not 2319, 2314, 3330, 3340, or 3350
- R15=3 Attempt to write on a read-only disk
- R15=4 Cylinder number not in range of user's disk
- R15=5 Virtual device is busy or has an interrupt pending

cc=2 Error condition. Register 15 contains one of the following:

- R15=5 Pointer to CCW string not doubleword-aligned.
- R15=6 SEEK/SEARCH arguments not within range of user's storage
- R15=7 READ/WRITE CCW is neither Read (06) nor Write (05)
- R15=8 READ/WRITE byte count=0
- R15=9 READ/WRITE byte count greater than 2048
- R15=10 READ/WRITE buffer not within user's storage
- R15=11 The value in R15, at entry, was not a positive number from 1 through 15, or was not large enough for the given CCW string.
- R15=12 Cylinder number on seek head was not the same number as on the first seek.

cc=3 Uncorrectable I/O error:

R15=13
CSW (8 bytes) returned to user
Sense bytes are available if user issues a SENSE command

DIAGNOSE Code X'1C' -- Clear Error Recording Cylinders

Execution of DIAGNOSE code X'1C' allows a user with privilege class F to clear the error recording data on disk. The DMKIOEFM routine performs the clear operation. The register specified as Rx contains a one-byte code value in the low-order byte as follows:

Code	Function
X'01'	Clear and reformat all error recording, leaving any frame records intact
X'02'	Clear and reformat all error recording cylinders, erasing both frame records and error records

DIAGNOSE Code X'20' -- General I/O

With DIAGNOSE code X'20', a virtual machine user can specify any valid CCW chain to be performed on a tape or disk device. No I/O interrupts are reflected to the virtual machine; the DIAGNOSE instruction is completed only when all I/O commands in the specified CCW chain are finished. The register specified as Rx contains the virtual device address. The Ry register contains the address of the CCW chain.

The CCWs are processed via DMKCCWTR through DMKGIOEX, providing full virtual I/O in a synchronous fashion (self-modifying CCWs are not permitted, however) to any virtual machine specified. Control returns to the virtual machine only after completion of the operation or detection of a fatal error condition. EREP support is provided for tape and DASD devices only; all other devices will present an error condition in the PSW to the virtual user. Condition codes and error codes are returned to the virtual system.

The condition codes and error codes returned are as follows:

cc=0 I/O completed with no errors

cc=1 Error condition. Register 15 contains the following:

- R15=1 Device is either not attached or the virtual channel is dedicated.
- R15=5 Virtual device is busy or has an interrupt pending.

cc=2 Exception conditions. Register 15 contains one of the following:

- R15=2 Unit exception bit in device status byte=1
- R15=3 Wrong length record detected.

cc=3 Error Condition:

- R15=13 A permanent I/O error occurred or an unsupported device was specified. The two rightmost positions of the user's Ry register contain the first two sense bytes

DIAGNOSE Code X'24' -- Device Type and Features

DIAGNOSE code X'24' requests CP to provide a virtual machine with identifying information and status information about a specified virtual device. The virtual machine must specify the virtual device for which information is requested. CP returns information about the virtual device and associated real device in the Rx, Ry, and Ry+1 registers. CP also provides a condition code identifying the specific device information returned to the virtual machine.

When a virtual machine issues DIAGNOSE code X'24', the Rx register must contain the virtual device address for which information is requested or the value negative 1 (-1). Specify -1 when the device is a virtual console whose address is unknown to the virtual machine.

When CP returns control to the virtual machine, the Ry, Ry+1, and Rx registers contain device information. The Ry register contains information about the virtual device and the Ry+1 register information about the real device. If -1 was specified and CP located the virtual console, the Rx register contains information about the virtual console.

CP obtains device information from three control blocks: virtual device information from the virtual device block (VDEVBLOK), and real

device information from the real device block (RDEVBLOK) and from NICBLOK. The following diagrams identify specific information returned by CP and show how to locate this information in the Rx, Ry, and Ry+1 registers. The symbolic names used in these diagrams are the symbolic names used with VDEVBLOK, RDEVBLOK, and NICBLOK in VM/370 Data Areas and Control Block Logic.

Rx Register

Byte 0	Byte 1	Byte 2	Byte 3
RDEVTMCD - or - NICTMCD			virtual device address

Symbolic Name	Meaning
RDEVTMCD - or - NICTMCD	Terminal code bits defining the type of console and the translate table the console is using. RDEVTMCD is for a local virtual console; NICTMCD for a remote 3270 virtual console

Ry Register

Byte 0	Byte 1	Byte 2	Byte 3
VDEVTYPC	VDEVTYPE	VDEVSTAT	VDEVFLAG

Symbolic Name	Meaning
VDEVTYPC	Virtual device type class
VDEVTYPE	Virtual device type
VDEVSTAT	Virtual device status
VDEVFLAG	Virtual device flags

Ry+1 Register

Byte 0	Byte 1	Byte 2	Byte 3
RDEVTYPC	RDEVTYPE - or - NICDTYPE	RDEVMDL - or - NICMDL	RDEVFTR - or - RDEVLLEN - or - NICLLEN

Symbolic Name	Meaning
RDEVTYPC	Real device type class
RDEVTYPE	Real device type
RDEVMDL	Real device model number
RDEVFTR	Real device feature code for a device other than a virtual console
RDEVLLEN	Current device line length for a local virtual console
NICDTYPE	Real device type for a remote 3270 virtual console
NICMDL	Real device model number for a remote 3270 virtual console
NICLLEN	Current device line length for a remote virtual console

The following chart lists the condition codes CP can return for DIAGNOSE code X'24', the meaning of each condition code, and the registers where data is returned.

If the condition code equals	This register contains information: Rx[1]	Ry	Ry+1[2]	Comments
0	X	X	X	Normal completion
1				Undefined
2	X	X		The virtual device exists but is not associated with a real device
3				Invalid device address or the virtual device does not exist

[1]The Rx register contains information only when DIAGNOSE code X'24' specifies a virtual console whose address in unknown.
[2]If Ry is register 15, CP returns only virtual device information: no information is returned in register Ry+1.

DIAGNOSE Code X'28' -- Channel Program Modification

DIAGNOSE code X'28' allows a virtual machine to correctly execute some channel programs modified after the Start I/O (SIO) instruction is issued and before the input/output operation is completed. The channel command word (CCW) modifications allowed are:

- A Transfer in Channel (TIC) CCW modified to a No Operation (NOP) CCW
- A TIC CCW modified to point to a new list of CCWs
- A NOP modified to a TIC CCW

When a virtual machine modifies a TIC CCW, it is modifying a virtual channel program. CP has already translated that channel program and is waiting to execute the real CCWs. The DIAGNOSE instruction, with code X'28', must be issued to inform CP of the change in the virtual channel program, so that CP can make the corresponding change to the real CCW before it is executed. In addition, when a NOP CCW is modified to point to a new list of CCWs, CP translates the new CCWs.

To be sure that the DIAGNOSE instruction is recognized in time to update the real CCW chain, the virtual machine issuing the DIAGNOSE instruction should have a high favored execution value and a low dispatching priority value. The CP SET command should be issued:

```
SET FAVORED xx

SET PRIORITY nn
```

where xx has a high numeric value and nn has a low numeric value. The virtual machine issuing the DIAGNOSE code X'28' must be in the supervisor mode at the time it issues the DIAGNOSE instruction.

When DIAGNOSE code X'28' is issued, the Rx register contains the address of the TIC or NOP CCW that was modified by the virtual machine. The Ry register contains the device address in bits 16 through 31. Rx and Ry cannot be the same register. The addresses specified in the Rx register, the new address in the modified TIC CCW, and the new CCW list to which the modified TIC CCW points must all be addresses that appear real to the virtual machine: CP knows these addresses are virtual, but the virtual machine thinks they are real.

The condition codes (cc) and completion codes are as follows:

cc=0 The real channel program was successfully modified; register 15 contains a zero.

cc=1 There was probably an error in issuing the DIAGNOSE instruction. Register 15 (R15) contains one of the following completion codes:

- R15=1 The same register was specified for Rx and Ry.
- R15=2 The device specified by the Ry register was not found.
- R15=3 The address specified by the Rx register was not within the user's storage space.
- R15=4 The address specified by the Rx register was not doubleword aligned.
- R15=5 A CCW string corresponding to the device (Ry) and address (Rx) specified was not found.
- R15=6 The CCW at the address specified by the Rx register is not a TIC nor a NOP, or the CCW in the channel program is not a TIC nor a NOP.
- R15=7 The new address in the modified TIC CCW is not within the user's storage space.
- R15=8 The new address in the modified TIC CCW is not doubleword aligned.

cc=2 The real channel program cannot be modified because a channel end or device end already occurred. Register 15 contains a 9. The virtual machine should restart the modified channel program.

DIAGNOSE Code X'2C' -- Return DASD Start of LOGREC

Execution of DIAGNOSE code X'2C' allows a user with privilege class C, E, or F to find the location on the disk of the error recording area, the number of error recording cylinders, and the location of the first error record.

The register specified as Rx contains a one-byte code in the low-order byte, indicating the function to be performed:

X'01' - Return the DASD location of the start of the error recording area, and the number of error recording cylinders.

X'02' - Return the HDRSTART value (DASD location of first error record).

X'04' - Return indication of whether there are frame records on the error recording cylinders.

On return to the issuer of DIAGNOSE '2C':

If code '01' is specified: Register Rx will contain the DASD location (in VM/370 control program internal format) of the start of the error recording area. Ry contains, in the low-order halfword, the number of error recording cylinders.

If code '02' is specified: Register Rx will contain the DASD location of the first error record (in CCPD format). The value actually points to the last frame record written, or record 2 if no frame records present.

If code '04' is specified: Register Ry will contain a X'02' in the low-order byte if frame records are present on the error recording cylinders; X'00' if no frame records present.

Note: Codes '02' and '04' may both be specified (code '06') on invoking DIAGNOSE. Both an Rx and Ry value must be specified.

DIAGNOSE Code X'30' -- Read One Page of LOGREC Data

Execution of DIAGNOSE code X'30' allows a user with privilege class C, E, or F to read one page of the system error recording area. The register specified as Rx contains the DASD location (in VM/370 control program internal format) of the desired record. The Ry register contains the virtual address of a page-size buffer to receive the data. The DMKRPAGT routine supplies the page of data. The condition codes returned are:

Condition Code	Meaning
0	Successful read, data available
1	End of cylinder, no data
2	I/O error
3	Invalid cylinder, outside recording area

DIAGNOSE Code X'34' -- Read System Dump Spool File

A user with privilege class C or E can read the system spool file by issuing a DIAGNOSE code X'34' instruction. The register specified as Rx contains the virtual address of a page-size buffer to receive the data. The Ry register, which must not be register 15, contains the virtual

address of the spool input card reader. Ry+1, on return, may contain error codes as follows:

Condition Code	Ry+1 Error Code	Meaning
0		Data transfer successful
1		End of file
2		File not found
3	4	Device address invalid
3	8	Device type invalid
3	12	Device busy
3	16	Fatal paging I/O error

The DMKDRDMP routine searches the system chain of spool input files for the dump file belonging to the user issuing the DIAGNOSE instruction. The first (or next) record from the dump file is provided to the virtual machine via DMKRPAGT and the condition code is set to zero. The dump file is closed via VM/370 console function CLOSE.

DIAGNOSE Code X'38' -- Read System Symbol Table

Execution of DIAGNOSE code X'38' causes the routine DMKDRDSY to read the system table into storage. The register specified as Rx contains the address of the page buffer to contain the symbol table.

DIAGNOSE Code X'3C' -- VM/370 Directory

Execution of DIAGNOSE code X'3C' allows a user to dynamically update the VM/370 directory. The register specified as Rx contains the first 4 bytes of the volume identification. The first two bytes of Ry contain the last 2 bytes of the volume identification. The routine DMKUDRDS dynamically updates the directory.

DIAGNOSE Code X'4C' -- Generate Accounting Cards for the Virtual User

This code can be issued only by a user with the account option (ACCT) in his directory.

Rx contains the virtual address of either a 24-byte parameter list identifying the "charge to" user, or a variable length data area that is to be punched into the accounting card. The interpretation of the address is based on a hexadecimal code supplied in Ry. If the virtual address represents a parameter list, it must be doubleword aligned; if it represents a data area, the area must not cross a page boundary. If Rx is interpreted as pointing to a parameter list and the value in Rx is zeros, the accounting card is punched with the identification of the user issuing the DIAGNOSE instruction.

Ry contains a hexadecimal code interpreted by DMKHVC as follows:

Code	Rx points to:
0000	a parameter list containing only a userid.
0004	a parameter list containing a userid and account number.
0008	a parameter list containing a userid and distribution number.
000C	a parameter list containing a userid, account number, and distribution number.
0010	a data area containing up to 70 bytes of user information to be transferred to the accounting card starting in column 9.

Note: If Ry contains X'0010', Ry cannot be register 15.

Ry+1 contains the length of the data area pointed to by Rx. If Rx points to a parameter list (Ry not equal to X'0010'), Ry+1 is ignored.

DMKHVC checks the VMACCOUN flag in VMPSTAT to verify that the user has the account option and if not, returns control to the user with a condition code of one.

If Ry contains a code of X'0010', DMKHVC performs the following checks:

- If the address specified in Rx is negative or greater than the size of the user's virtual storage, an addressing exception is generated.
- If the combination of the address in Rx and the length in Ry+1 indicates that the data area crosses a page boundary, a specification exception is generated.
- If the value in Ry+1 is zero, negative, or greater than 70, a specification exception is generated.

If both the virtual address and the length are valid, DMFREE is called to obtain storage for an account buffer (ACNTBLOK) which is then initialized to blanks. The userid of the user issuing the DIAGNOSE instruction is placed in columns 1 through 8 and an accounting card identification code of "C0" is placed in columns 79 and 80. The user data pointed to by the address in Rx is moved to the accounting card starting at column 9 for a length equal to the value in Ry+1. A call to DMKACOQU queues the ACNTBLOK for real output. If a real punch is available, DMKACOPU is called to punch the card; otherwise, the buffer is stored in main storage until a punch is free. DMKHVC then returns control to the user with a condition code of zero.

If Ry contains other than a X'0010' code, control is passed to DMKCPV to generate the card. DMKCPV passes control to DMKACO to complete the "charge to" information; either from the User Accounting Block (ACCTBLOK), if a pointer to it exists, or from the user's VMBLOK. DMKCPV then punches the card and passes control back to DMKHVC to release the storage for the ACCTBLOK, if one exists. DMKHVC then checks the parameter list address for the following conditions:

- If zero, control is returned to the user with a condition code of zero.
- If invalid, an addressing exception is generated.
- If not aligned on a doubleword boundary, a specification exception is generated.

For a parameter list address that is nonzero and valid, the userid in the parameter list is checked against the directory list and if not found, control is returned to the user with a condition code of two. If the function hexadecimal code is invalid, control is returned to the user with a condition code of three. If both userid and function hexadecimal code are valid, the User Accounting Block (ACCTBLOK) is built and the userid, account number, and distribution number are moved to the block from the parameter list or the User Machine Block belonging to the userid in the parameter list. Control is then passed to the user with a condition code of zero.

DIAGNOSE Code X'50' -- Save the 370X Control Program Image

DIAGNOSE code X'50' (Privilege class A, B, or C only) invokes the CP module DMKSNC to (1) validate the parameter list and (2) write the page-format image of the 370X control program to the appropriate system volume.

When a 370X control program load module is created, the CMS service program SAVENCP builds a communications controller list (CCPARM) of control information. It passes this information to CP via a DIAGNOSE code X'50'.

The register specified as Rx contains the virtual address of the parameter list (CCPARM). The Ry register is ignored on entry.

Upon return, the Ry register contains the following error codes:

Code	Meaning
044	'ncpname' was not found in system name table.
171	System volume specified not currently available.
178	Insufficient space reserved for program and system control information.
179	System volume specified is not a CP-owned volume.
435	Paging error while writing saved system.

DIAGNOSE Code X'54' -- Control The Function of the PA2 Function Key

DIAGNOSE code X'54' controls the function of the PA2 function key. The PA2 function key can be used either to simulate an external interrupt to a virtual machine or to clear the output area of a display screen.

The function performed depends upon how Rx is specified when DIAGNOSE code X'54' is issued. If Rx contains a nonzero value, the PA2 key simulates an external interrupt to the virtual machine. If Rx contains a value of zero, the PA2 key clears the output area of the display screen.

The external interrupt is simulated only when the display screen is in the VM READ, HOLD, or MORE status and the TERMINAL APL ON command has been issued.

DIAGNOSE Code X'58' -- 3270 Virtual Console Interface

Execution of DIAGNOSE code X'58' allows a virtual machine to display large amounts of data on a 3270 in a very rapid fashion. The interface can display the entire 3270 screen with one write operation instead of 22 writes (one for each line in the output area of a 3270 screen).

The register specified as Rx contains the address of the console CCW string. The Ry register contains (in bits 16 through 31) the device address of the virtual console.

To specify the special display CCW, use the following assembler language instructions:

```
DS 0D
DC X'19', AL3 (dataddr), AL1(flags), AL1(ctl), AL2(count)
```

where:

dataddr is the beginning address of the data to be displayed.

flags is the standard CCW flag field with the suppress incorrect length indication (SLI) bit on.

ctl is a control byte that indicates the starting output display line. If the high order bit is on, the entire 3270 output display area is erased before the new data is displayed. A value of X'FF' clears the screen, but writes nothing.

count is a two byte field indicating the number of bytes to be displayed. The maximum number of bytes for the 3278 Model 2A is 1440. For other types it is 1760.

When the DIAGNOSE is executed with a valid CCW string, a buffer (whose length is the number of bytes specified by count) is built in free storage. The data pointed to by dataddr is loaded into the buffer. Data chaining may be specified in the CCW to link noncontiguous data areas; however, command chaining is an end of data indication for the current buffer.

Using the starting output line (ctl) and the number of bytes of output (count), CP checks that the data will fit on the screen. CP then does the display. A zero condition code indicates the I/O operation completed successfully; a nonzero condition code indicates an I/O error occurred.

Note: An I/O error occurs when the display screen is placed in MORE status and the PA2 key is pressed to allow screen display.

DIAGNOSE Code X'5C': Error Message Editing

Execution of DIAGNOSE code X'5C' causes the editing of an error message according to the user's setting of the EMSG function:

Rx contains the address of the message to be edited.

Ry contains the length of the message to be edited.

DMKHVC tests the VMMLEVEL field of the VMBLOK and returns to the caller with Rx and Ry modified as follows:

VMMLEVEL		Registers on Return	
VMMCODE	VMMTEXT	Rx	Ry
ON	ON	no change	no change
ON	OFF	no change	10 (length of code)
OFF	ON	pointer to text part of message	length of text alone
OFF	OFF	N/A	0

Note: DIAGNOSE code X'5C' does not write the message; it merely rearranges the starting pointer and length. For CMS error messages, a console write is performed following the DIAGNOSE unless Ry is returned with a value of 0.

DIAGNOSE Code X'60' - Determining the Virtual Machine Storage Size

Execution of DIAGNOSE code X'60' allows a virtual machine to determine its size. On return, the register specified as Rx contains the virtual machine storage size.

DIAGNOSE Code X'64' - Finding, Loading, and Purging a Named Segment

Execution of DIAGNOSE code X'64' controls the linkage of discontiguous saved segments. The type of linkage that is performed depends upon the function subcode in the register specified as Ry.

Subcode		Function
X'00'	LOADSYS	-- Loads a named segment in shared mode
X'04'	LOADSYS	-- Loads a named segment in nonshared mode
X'08'	PURGESYS	-- Releases the named segment from virtual storage
X'0C'	FINDSYS	-- Finds the starting address of the named segment

The register specified as Rx must contain the address of the name of the segment. The segment name must be 8 bytes long, left justified, and padded with trailing blanks.

The LOADSYS Function

When the LOADSYS diagnose function is executed, CP finds the system name table entry for the named system and builds the necessary page and swap tables. Two sets of page and swap tables, one for each processor, are built for each shared segment in attached processor mode unless the named system was defined as unprotected. CP releases all the virtual pages of storage that are to contain the named segment and then loads the segment in those virtual pages. When the LOADSYS function is executed, CP expands the virtual machine size dynamically, if necessary. CP also expands the segment tables to match any expansion of virtual storage.

When LOADSYS executes successfully, the address of where the named segment was loaded is returned in the register specified as Rx. When the LOADSYS function loads a segment in shared mode, it resets instruction and branch tracing if either was active.

After a LOADSYS function executes, the storage occupied by the named segment is addressable by the virtual machine, even if that storage is beyond the storage defined for the virtual machine. However, any storage beyond that defined for the virtual machine and below that defined for the named segment is not addressable. Figure 15 shows the virtual storage that is addressable before and after the LOADSYS function executes.

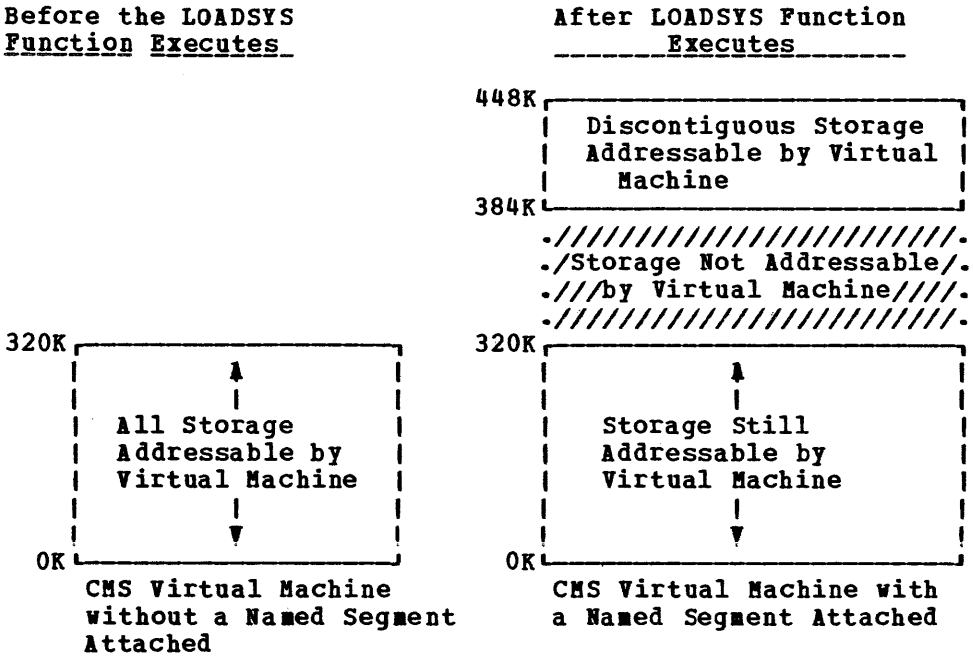


Figure 15. Addressable Storage Before and After a LOADSYS Function

When you save a named segment that is later loaded by the LOADSYS function, you must be sure that the addresses at which segments are saved are correct and that they do not overlay required areas of storage in the virtual machine. This is crucial because the LOADSYS function invokes the PURGESYS function before it builds the new page and swap tables. CP purges all saved systems that are overlayed in any way by the saved system it is loading.

A condition code of 0 in the PSW indicates that the named segment was loaded successfully; the Rx register contains the load address.

A condition code of 1 in the PSW indicates the named segment was loaded successfully within the defined storage of the virtual machine. The Rx register contains the address at which the named segment was loaded. The Ry register contains the ending address of the storage released before the named segment was loaded.

Note: CMS only allows named segments to be attached beyond the defined size of the virtual machine.

A condition code of 2 in the PSW indicates the LOADSYS function did not execute successfully. Examine the return code in the Ry register to determine the cause of the error.

Return Code	Meaning
44	Named segment does not exist
177	Paging I/O errors

The PURGESYS Function

When the PURGESYS function is executed; CP releases the storage, and associated page and swap tables, that were acquired when the

corresponding LOADSYS function was executed. If the storage occupied by the named segment was beyond the defined virtual machine storage size, that storage is no longer addressable by the virtual machine.

When a PURGESYS function is executed for a segment that was loaded in nonshared mode, the storage area is cleared to binary zeros. If PURGESYS is invoked for a named segment that was not previously loaded via LOADSYS, the request is ignored.

A condition code of 0 in the PSW indicates successful completion.

A condition code of 1 in the PSW indicates that the named segment was not found in the virtual machine.

A condition code of 2 in the PSW and a return code of 44 in the Ry register indicate that the named segment either does not exist or was not previously loaded via the LOADSYS function.

The FINDSYS Function

When the FINDSYS function is executed, CP checks that the named segment exists and that it has not been loaded previously.

A condition code of 0 in the PSW indicates that the named segment is already loaded. The address at which it was loaded is returned in the register specified as Rx and its highest address is returned in the Ry register.

A condition code of 1 in the PSW indicates that the named segment exists but has not been loaded. In this case, the address at which the named segment is to be loaded is returned in the register specified as Rx and the highest address of the named segment is returned in the Ry register.

A condition code of 2 in the PSW indicates the FINDSYS function did not execute successfully. Examine the return code in the Ry register to determine the error that occurred.

Return Code	Meaning
44	Named segment does not exist
177	Paging I/O errors

DIAGNOSE Code X'68' -- Virtual Machine Communication Facility (VMCF)

The DIAGNOSE code X'68' is used by a virtual machine to initiate a subfunction of the Virtual Machine Communication Facility (VMCF). The general register specified as Rx contains the virtual address, doubleword aligned, of a parameter list (VMCPARM). One of the entries in this parameter list is a subfunction code, specifying the particular request being initiated. The subfunctions and their codes are:

Subfunction	Code
AUTHORIZE	X'0000'
UNAUTHORIZE	X'0001'
SEND	X'0002'
SEND/RECV	X'0003'
SENDX	X'0004'
RECEIVE	X'0005'
CANCEL	X'0006'
REPLY	X'0007'
QUIESCE	X'0008'
RESUME	X'0009'
IDENTIFY	X'000A'
REJECT	X'000B'

A description of all the fields of the VMCPARM is contained in the VM/370 System Programmer's Guide.

The general register specified as Ry will contain the return code upon completion of DIAGNOSE X'68' or the detection of an error condition. The return codes are contained in the VM/370 System Programmer's Guide.

Rx and Ry can be any general register, R0 through R15. They may also be the same register.

DIAGNOSE Code X'74' -- Load 3800 Named System Into Virtual Storage

DIAGNOSE code X'74' allows a virtual machine to load a 3800 named system into virtual storage beginning at a specified virtual address and to take a copy of an image library and save it in a specified named system. A named system is one that contains 3800 character arrangement tables, copy modifications, and file control blocks (FCBs). These can be referenced by name, and the data can be obtained when the file referencing them is about to print on a 3800 printer. The new data in the named system is used to load into the 3800 the next time NAMED SYSTEM is specified as the IMAGELIB parameter in a START command for a 3800 printer.

At system generation time, the NAME3800 macro instruction establishes the name of the 3800 system in the DMKSNT module. The active named system for a particular 3800 printer will be in its RDEVBLOK.

The GENIMAGE command creates the image library in virtual storage. This is done by interfacing with the OS IEBIMAGE utility program (DMKIMG). The IMAGELIB command saves the image library and issues the DIAGNOSE code X'74' to establish the named system.

When using DIAGNOSE code X'74', the registers specified by the user as Rx and Rx+1 must contain the eight-character name of the system desired, left-justified and padded with trailing blanks.

The register specified by the user as Ry must contain the virtual address at which to start loading or saving the storage image of an image library.

The register specified by the user as Ry+1 must contain a one-byte code in the high-order byte indicating the function to be performed as follows:

X'00' - LOAD operation
X'04' - SAVE operation

The low-order bytes of register Ry+1 must contain the number of bytes to load or save.

If either Rx or Ry is specified as register 15, or if the virtual address specified in Ry is not on a page boundary, a specification exception results and the program terminates.

An addressing exception results if the end of the area to be loaded or saved extends beyond the end of the user's virtual storage. A privileged operation results if the user does not have privileged class A, B, or C.

Register Ry contains a return code upon completion of processing as follows:

Condition Code	Meaning
X'00'	Load or save successfully performed
X'04'	Named system not found
X'08'	Named system currently active
X'0C'	Volid for system not CP-owned
X'10'	Volid for system not mounted
X'14'	Too many bytes to load or save. Residual Byte count is in register Ry+1
X'18'	Paging error during load or save

DIAGNOSE Code X'78' -- MSS Support

DIAGNOSE code X'78' is used to communicate between a virtual machine and CP for MSS support. A subfunction code is placed in the Ry register. The specific subfunction codes and their meanings are:

Subfunction Code	Meaning
X'00'	The virtual machine is running OS/VS with MSS support and is now ready to communicate with both VM/370 and the mass storage control (MSC). The Rx field of the instruction contains the number of a register that contains the virtual device address of the MSS communication device.
X'04'	The virtual machine is ready to process an MSS request. The request, in the form of an MSSCOM control block, is to be placed at the virtual machine address indicated by the register in the Rx field of the instruction.
X'08'	The MSS request, in the form of an MSSCOM control block, located at the virtual machine address contained in register Rx, has been processed by the MSC.
X'10'	The virtual machine is no longer able to process MSS requests.

DIAGNOSE Code X'84' -- Directory Update In-Place

DIAGNOSE code X'84' can be used to make changes to the online directory if the change causes no expansion to the entry and when the user wants the change to go into effect immediately. This interface is provided for the privilege class B virtual machine that owns the directory. Validity checking is performed on update parameters and the specified user's directory is updated in storage and written on the current DASD copy. If any errors are encountered, either in specifications or in DASD I/O, the update in-place is not performed and an error condition is passed to the invoking virtual machine.

General register Rx points to a variable length parameter list. Register Ry contains the length of the parameter list, in bytes. The parameter list contains fixed common fields and variable fields as shown below:

Field	Name	Function
Common	UUSERID	Userid to be updated
	UCURPASS	Current logon password
	UOP	Update operation to perform
	UCMMONSZ	Length of common fields
Variable	UNEWPASS	New logon password
	USTORAGE	New storage size
	UPRIV	Up to eight privilege classes
	UPRIOR	New priority (a number between 0 and 99)
	UEDITCH	Edit characters (LE, LD, CD, ES)
	UIPL	New IPL name
	UACCOUNT	New account data
	UDISTRIB	New distribution data
	UMDISKAD	New minidisk address
	UMDISKMD	New minidisk link mode
	UMDISKRP	New minidisk read password
	UMDISKWP	New minidisk write password
	UMDISKMP	New minidisk multiple password
	UOPTIONS	Up to 9 options

The variable field chosen is positioned after the UOP field. Only one variable field can be used at one time. A separate DIAGNOSE code X'84' must be issued for each variable function desired.

If any errors are found, the condition code is set to 1 and register Ry may contain one of the following codes:

Return Code	Meaning
10	Error in DMKRPAPT when writing object DASD
11	Error in DMKRPAPT when writing paging DASD
20	Error during 'TRANS' of UDIR page
21	No UMAC address in UCNTRL
22	Error during 'TRANS' of UMAC page
23	No UDEV address in UCNTRL
24	No UDEV address in UCNTRL
25	Error during 'TRANS' of UDEV page
26	UDEV block not found
27	Object DASD not synchronous with DMKSYSPL
28	Operation invalid
30	Userid not found
31	Logon password mismatch
40	Storage exceeds maximum allowed
41	Maximum storage greater than 16M
42	No sign after packing new size

Return Code	Meaning
43	Invalid bytes in storage data
50	Privilege operand all blanks
51	No valid privilege classes in operand
52	Error in accumulated privilege value
53	Invalid data in privilege field
60	Invalid data in priority field
61	Priority field all blanks
62	No sign after packing new priority
63	Priority greater than maximum
70	Invalid option
71	No fence of X'FF's at end of parameter list
72	Invalid accumulated option values, signifying an option error
80	Invalid MDISK address in parameter list
81	Invalid link mode

If no errors are found, the condition code is set to zero.

I/O Interruption

I/O interruptions from completed I/O operations initiate various completion routines and the scheduling of further I/O requests. The I/O interruption handling routine also gathers device sense information.

Machine Check Interruption

When a machine check occurs, CP Recovery Management Support (RMS) gains control to save data associated with the failure for FE maintenance. RMS analyzes the failure and determines the extent of damage.

Damage assessment results in one or more of the following actions being taken:

- System termination
- Attached processor varied offline (system converts to uniprocessor mode)
- Virtual user running at the time of error is terminated
- Refreshing of damaged information with no effect on system configuration
- Refreshing of damaged information with the defective storage page removed from further system use
- Error recording only for certain soft machine checks

The system operator is informed of all actions taken by the RMS routines. When a machine check occurs during VM/370 startup (before the system is set up well enough to permit RMS to operate successfully), the processor goes into a disabled wait state and places a completion code of X'00B' in the leftmost bytes of the current PSW.

SVC Interruption

When an SVC interruption occurs, the SVC interruption routine (DMKSVCIN) is entered. If the machine is in the problem state, DMKSVCIN takes the following action:

- If the interruption was the result of an ADSTOP (SVC code X'B3'), the message ADSTOP AT XXXXX is sent to the user's terminal, the overlaid instruction is replaced, and the virtual machine is placed in console function mode (CP mode) via DMKCFMBK.

- If the interruption was the result of an error recording interface (SVC 76), DMKSVC checks for valid parameters and passes control to DMKVER to convert virtual device addresses in the error record to real device addresses. The actual recording is accomplished in DMKIOE and DMKIOF. If recording is not possible, the interrupt is reflected back to the virtual machine.

- If the virtual machine's page 0 was not in real storage, then all general and floating-point registers are saved, the user's VMBLOK is flagged as being in an instruction wait, and control is transferred (via GOTO) to DMKPRGRF to reflect the interruption to the virtual machine.

- If the virtual machine's page 0 is in main storage, an appropriate SVC old PSW is stored in the user's page 0 and the interruption is reflected to the virtual machine, bypassing unnecessary register saving (fast reflection). If the new virtual PSW indicates a mode or enablement change, all registers are saved in the VMBLOK and control is transferred to DMKDSPB for PSW validation.

If the machine is in the supervisor state, the SVC interruption code is determined and a branch is taken to the appropriate SVC interruption handler.

SVC 0
Impossible condition or terminal error. The SVCDIE routine initiates an abnormal termination by using the DMKDMPDK routine.

SVC 4
Reserved for IBM use.

SVC 8
A link request that transfers control from the calling routine to the routine specified by register 15. The SVCLINK routine sets up a new save area, and then saves the caller's base register in register 12 and save area address in register 13, and the return address (from the SVCOPSW) in the new save area. If the called routine is within the resident CP nucleus, SVCLINK places its address in register 12 and branches directly to the called routine. If the called routine is in a pageable module, a TRANS macro is performed for register 12 to ensure that the page containing the called routine is in storage. Upon return from the TRANS execution, the real address of the pageable routine is placed in register 12 and SVCLINK branches to the called routine. The real storage location of DMKCPE is the end of the resident CP nucleus. Any modules loaded at a higher real storage address are defined as pageable modules. If bit zero of register 15 is on when DMKSVC is entered, then the caller has requested AFFINITY. DMKSVC turns on a bit in the save area passed to the caller to indicate that control is to be returned to the caller on the same processor on which it was running before issuing the SVC. It is not ensured that control will be retained by the initiating processor throughout the called operation, but only that final return will occur on the initiating processor.

SVC 12
A return request that transfers control from the called routine to the calling routine). The SVCRET routine is invoked. If the routine that issued the SVC 12 is pageable, then DMKPTRUL is called to unlock the page. SVCRET then restores registers 12 and 13 (addressability and save area address saved by SVCLINK), places the user's return address (also saved in this area) back into the SVCOPSW, and returns control to the calling routine by loading the SVCOPSW.

SVC 16
Releases current save area from the active chain (removes linkage pointers to the calling routine). The SVCRLSE routine releases the current save area by placing the address of the next higher save area in register 13 and returns control to the current routine by loading the SVCOPSW. This SVC is used by second level interrupt handlers to bypass returning the first-level handler under specific circumstances. The base address field (register 12) in the save area being released is examined to determine if the bypassed routine is in a pageable module. If so, DMKPTRUL is called to unlock the page.

SVC 20
Obtain a new save area. The SVCGET routine places the address of the next available save area in register 13 and the address of the previous save area in the save area pointer field of the current save area.

SVC 24
In attached processor mode, SVC 24 causes the instructions following the SVC to be executed by the main processor. This SVC is used only via the SWITCH macro to force processing to continue on the main processor (the processor capable of performing I/O). If the SWITCH macro determines that the code is currently running on the main processor then the SVC is not issued.

There are 35 save areas initially set up by DMKCPINT for use by the SVC linkage handlers. If all the save areas are used, the linkage handlers call DMKFREE to obtain additional save areas.

External Interruption

TIMER INTERRUPTION

If DMKPSAEX is entered because of a timer interruption, the state of the machine must be determined. If the machine was in wait state, control is transferred to DMKDSPCH, and the machine stays idle until another interruption occurs. If the machine is in problem state, the address of the current user's VMBLOK is obtained from RUNUSER. The user's current PSW (VMPSW) is updated from the external interruption old PSW, the address of the current VMBLOK is placed in register 11, and control is transferred to DMKDSPCH. For additional information about timers, see "Virtual Timer Maintenance."

EXTERNAL INTERRUPTION

If DMKPSAEX is entered because the operator pressed the console interrupt button (INTERRUPT), a CPEXBLOK is stacked to do the following:

- Reference the current system operator's VMBLOK (DMKSYSOP).
- Disconnect this virtual machine.

The operator can now log on from another terminal. Pressing the console interrupt button activates an alternate operator's console.

Note: If this interrupt comes from the attached processor, it is ignored.

For a description of the processing of the external interruption command, refer to module DMKCPB in Section 2.

See "Multiprocessor External Interrupts" for a discussion of external interrupts that occur in attached processor mode.

EXTENDED VIRTUAL EXTERNAL INTERRUPTIONS

To reflect external interruptions to a virtual machine, DMKDSPE queues an XINTBLOK on a chain pointed to by VMPXINT in the VMBLOK. The XINTBLOKs are chained sequentially by the XINTSORT field that contains the collating number of the pending interruption. If more than one interruption has the same collating number, the interruption codes are ORed together in the XINTCODE field for possible simultaneous reflection.

When a virtual machine is enabled for external interruptions, the XINTBLOK queue for that machine is searched for an eligible block. An XINTBLOK is eligible for reflection if one or more bits of the XINTMASK field match the bits in the rightmost halfword of control register 0. If the interruption was an interruption such as CPU timer or clock comparator, the block is left chained because reflection does not reset these interruptions. If the reflected interruption(s) does not represent all those coded in the XINTMASK field, the block is left chained and only the interruptions that were reflected are reset. In all other conditions, the XINTBLOK is unchained and returned to free storage.

A special external interrupt, code X'4001' notifies a virtual machine of a pending Virtual Machine Communication Facility request. The XINTBLOK for this interrupt is set up with an XINTSORT field of X'7FFFFFFF', the lowest priority.

System Support

FREE STORAGE MANAGEMENT

During its execution, CP occasionally requires small blocks of storage that are used for the duration of a task. CP obtains this storage from the free storage area. The free storage area is divided into various size subpools. The requester informs the free storage manager of the size of the block required and the smallest available subpool that fulfills the request is allocated to the requester. When the block is no longer needed, the requester informs the free storage manager and CP returns the block to free storage.

If the request for free storage cannot be fulfilled, the free storage manager requests the temporary use of a page of storage from the dynamic paging area. If a page is obtained, the page is chained to the free storage area and used for that purpose until it is no longer needed and subsequently returned to the dynamic paging area.

If the request for a page cannot be fulfilled, the requester waits until free storage becomes available.

STORAGE PROTECTION

VM/370 provides both fetch and store protection for real storage. The contents of real storage are protected from destruction or misuse caused by erroneous or unauthorized storing of fetching by the program. Storage is protected from improper storing or from both improper storing and fetching, but not from improper fetching alone.

When the processor accesses storage, and protection applies, the protection key of the current PSW is used as the comparand. The protection key of the processor is bit positions 8-11 of the PSW.

If the processor access is prohibited because of a protection violation, the operation is suppressed or terminated, and a program interruption for a protection exception takes place.

When the reference is made to a channel, and protection applies, the protection key associated with the I/O operation is used as the comparand. The protection key for an I/O operation is in bit positions 0-3 of the CAW and is recorded in bit positions 0-3 of the CSW stored as a result of an I/O operation. If channel access is prohibited, the CSW stored as a result of the operation indicates a protection-check condition.

When a storage access is prohibited because of a store protection violation, the contents of the protected location remain unchanged. If a fetch protection violation occurs, the protected information is not loaded into an addressable register, moved to another storage location, or provided to an I/O device.

To use fetch protection, a virtual machine must execute the set storage key (SSK) instruction referring to the data areas to be protected, with the fetch protect bit in the key. VM/370 subsequently:

1. Checks for a fetch protection violation when handling privileged and nonprivileged instructions.

2. Saves and restores the fetch protection bit (in the virtual storage key) when writing and recovering virtual machine pages from the paging device.

3. Checks for a fetch protection violation on a write CCW (except for spooling or console devices).

A special case of storage protection occurs when the CMS nucleus resides in a protected shared segment. The CMS nucleus may be protected and still be shared by many CMS users. After a virtual machine has used a protected shared segment, the pages are checked for changes. If any pages have been changed, the user gets placed in console function mode after receiving error message DMKVMA456W, and the changed page is returned to CP free storage.

EXECUTING THE PAGEABLE CONTROL PROGRAM

Calls to pageable routines are recognized at execution time by the SVC 8 linkage manager in DMKSVC. For every SVC 8, the called address (in the caller's GPR15) is tested to see if it is within the resident nucleus. If it is less than DMKCPEND and greater than DMKSLC, the called routine's base address is placed in GPR12 and control is passed to the called routine in the normal way. However, if the called address is above DMKCPEND or below DMKSLC, the linkage manager issues a TRANS macro, requesting the paging manager to locate and, if necessary, page-in the called routine. The TRANS is issued with LOCK option. Thus, the lock count associated with the called routine's real page indicates the responsibility count of the module.

- When the module is called, the count is incremented.
- When the routine exits via SVC 12, the count is decremented.

When the count reaches zero, the pageable routine is unlocked and is eligible to be paged out of the system. However, because all CP pageable modules are reenterable, the page is never swapped out, but when the page is stolen, it is placed directly on the free page list.

Because unlocked pageable routines participate in the paging process in a manner similar to user virtual storage pages, the least recently used approximation used by page selection tends to make highly used control program routines, even when not locked, remain resident. The called routine is locked into real storage until it exits. Thus, it can request asynchronously scheduled function, such as I/O or timer interrupts, as long as it dynamically establishes the interruption return address for the requested operation and does not give up control via an EXIT macro prior to receiving the requested interruption.

Addressability for the module, while it is executing, is guaranteed because the CALL linkage loads the real address of the paged module into GPR12 (the module base register) prior to passing control. If all addressing is done in a base/displacement form, the fact that the module is executing at an address different from that at which it was loaded is not apparent. Although part of CP is pageable, it never runs in relocate mode. Thus, the processor is not degraded by the DAT feature being active, and no problems occur because of handling disabled page faults.

SYSTEM SUPPORT MODULES

The system support modules provide CP with several common functions for data conversion and control block scanning and verification. Most of the routines are linked to via the BALR option of the CALL macro, and make use use of the BALRSAVE and TEMPSAVE workareas in DMKPSA. Two exceptions are the virtual and real I/O control block scan routines DMKSCNVU and DMKSCNRU. These routines do not alter the contents of the BALRSAVE area, and hence may be called by another low-level BALR routine.

CONTROL REGISTER USAGE

Every IBM System/370 processor provides the program with 16 logical control registers (logical registers since the number that are active depends on the features installed in the machine at any one time) that are addressable for loading and storing from basic control (BC) mode. VM/370 provides only a single control register, control register zero, for normal virtual machines, and for processing systems that do not require the full set of registers (for example, CMS, DOS, or other operating systems for System/360).

Any user whose virtual machine operating system requires the use of control registers other than control register zero, can request the full set of 16 registers by specifying the ECMODE option in the VM/370 directory entry for his virtual machine.

A virtual machine, which utilizes any System/370 features that use the control registers, requires the ECMODE option. Some of these features are expanded timer support of the System/370 CPU timer, clock comparator, etc., the virtual relocate mode and its instructions, RRB, LRA, PTLB, virtual monitor calls, virtual Program Event Recording (PER), etc.

RESTRICTIONS AND CONVENTIONS FOR PAGEABLE CP MODULES

Pageable CP modules must observe the following restrictions and conventions when they are designed and coded:

- The module must be entered by the standard SVC 8 CALL linkage. Modules entered by BALR or GOTO cannot be pageable. The module must return to its caller by SVC also.

- The module cannot contain any A- or V-type address constants that point to locations within itself or within other pageable modules, and it cannot contain any CCWs that contain data addresses within themselves. The only exceptions are address constant literals generated as the result of calls to other modules (because these addresses are dynamically relocated at execution time, they must be resolved by the loader to the loaded address of the called module) and a pageable module that locks itself into storage. In practice, this restriction means that data or instructions within the pageable routine must be referenced via base/displacement addressing, and the address in register 15 for a CALL may not be generated by a LOAD ADDRESS instruction.

- The pageable module must be no more than 4096 bytes in length.

If the three above design and coding restrictions are adhered to, the CP module can be added to the existing pageable nucleus modules by utilizing the service routine, VMFLOAD, which is described in "VM/370 Maintenance Procedures" of the VM/370 Service Routines Program Logic. Additional information can be found in the VM/370 Planning and System Generation Guide.

Executable Resident Modules

DMKBSC	DMKGRF	DMKPRG	DMKSSS
DMKCCH	DMKGRT	DMKPRV	DMKSTK
DMKCCW	DMKHVC	DMKPSA	DMKSVC
DMKCFM	DMKIOE	DMKPTR	DMKTMR
DMKCNS	DMKIOS	DMKQCN	DMKTRK
DMKCVT	DMKLOC	DMKRGA	DMKUNT
DMKDAS	DMKLOK	DMKRGB	DMKVAT
DMKDGD	DMKMCH	DMKRNH	DMKVCN
DMKDMP	DMKMCT	DMKRPA	DMKVIO
DMKDSB	DMKMSW	DMKRSP	DMKVMA
DMKDSP	DMKOPR	DMKSCH	DMKVSI
DMKEXT	DMKPAG	DMKSCN	DMKVSP
DMKFRE	DMKPGT		

Executable Pageable Modules

DMKACO	DMKCPB	DMKDIB	DMKNEM	DMKTRC
DMKALG	DMKCPI	DMKDRD	DMKNES	DMKTRD
DMKAPI	DMKCPS	DMKEIG	DMKNET	DMKTRM
DMKATS	DMKCPU	DMKERM	DMKNLD	DMKUDR
DMKBLD	DMKCPV	DMKGIO	DMKNLE	DMKUDU
DMKCDB	DMKCQG	DMKHVD	DMKPGS	DMKUSO
DMKCDM	DMKCQH	DMKIOC	DMKRSE	DMKVCA
DMKCDS	DMKCQP	DMKIOF	DMKSAV	DMKVCH
DMKCFC	DMKCQR	DMKIOG	DMKSEP	DMKVDA
DMKCFD	DMKCQY	DMKISM	DMKSEV	DMKVDC
DMKCFG	DMKCSB	DMKJRL	DMKSIX	DMKVDD
DMKCFH	DMKCSO	DMKLNK	DMKSNC	DMKVDE
DMKCFO	DMKCSP	DMKLOG	DMKSPL	DMKVDR
DMKCFP	DMKCSQ	DMKLOH	DMKTAP	DMKVDS
DMKCFS	DMKCST	DMKMCC	DMKTCS	DMKVER
DMKCFT	DMKCSU	DMKMID	DMKTDK	DMKVMC
DMKCKP	DMKCSV	DMKMNI	DMKTHI	DMKVMI
DMKCKS	DMKDEF	DMKMON	DMKTRA	DMKWRM
DMKCLK	DMKDIA	DMKMSG		

Figure 16. Executable Modules

DATA AREA MODULES

In addition to the executable resident and pageable modules (see Figure 16), there are certain modules that only contain data areas and do not contain executable code. These modules are:

Resident Module	Contents
DMKCPE	Defines the end of the CP nucleus
DMKGRW	CCW's and data for 3278 model 2A
DMKRIO	I/O device blocks
DMKSYS	System constants
DMKTBL	Terminal translate table

Pageable Module	Contents
DMKBOX	Output separator table
DMKBTS	Bootstrap routines for 3705
DMKEMA	Error message data module
DMKEMB	Error message data module
DMKEMC	Error message data module
DMKFCB	3203 and 3211 Forms Control Buffer (FCB) load tables
DMKSNT	System name table
DMKSYM	System symbol table
DMKUCB	3211 Universal Character Set Buffer (UCSB) load tables
DMKUCS	1403 Universal Character Set (UCS) load tables
DMKTBM	Terminal translate tables
DMKVCC	3203 Universal Character Set Buffer (UCSB) load tables

VIRTUAL TIMER MAINTENANCE

The System/370 with EC mode provides the system user (both real and virtual) with four timing facilities. They are:

- The interval timer at main storage location X'50'
- The time-of-day clock
- The time-of-day clock comparator
- The CPU timer

Real Timing Facilities

Before describing how CP maintains these timers for virtual machines, it is necessary to review how VM/370 uses the timing facilities of the real machine.

1. The location X'50' interval timer is used only for time-slicing. The value placed in the timer is the maximum length of time that the dispatched virtual machine is allowed to execute.

   Because the BLIP function of CMS uses the interval timer (location X'50'), the use of STIMER can cause extra blips at the user's terminal. To avoid extra blips, issue the CMS command SET BLIP OFF.

2. The time-of-day clock is used as a time stamp for messages and enables the scheduler to compute elapsed in-queue time for the dispatching priority calculation.

3. The time-of-day clock comparator facility is used by CP to schedule timer-driven events for both control program functions and for virtual machines. A stack of comparator requests is maintained and as clock comparator interrupts occur, the timer request blocks are stacked for the dispatcher via calls to DMKSTKIO.

4. The processor timer facility performs three functions:

   - Accumulates CP overhead
   - Detects in-queue time slice end
   - Simulates virtual processor timer

   The accumulation of CP overhead is accomplished as follows. The VMTTIME field in the VMBLOK contains the total CP overhead incurred by the virtual machine; it is initialized to the maximum positive number in a doubleword, X'7FFFFFFF FFFFFFFF'. Whenever CP performs

a service for a virtual machine, GR 11 is loaded with the address of the VMBLOK and the current value in VMTTIME is placed in the processor timer. When CP is finished with the service for that virtual machine the processor timer, which has been decremented by the amount of processor time used, is stored back into VMTTIME. GR 11 is then loaded with a new VMBLOK pointer and the processor timer is set from the new VMTTIME field. The amount of CP overhead for a given virtual machine at any point in time is the difference between the maximum integer and the current value in the VMTTIME field.

Since VMTTIME only accounts for supervisor state overhead, detection of in-queue time slice end is performed by the processor timer when the virtual machine is dispatched in the problem state. The VMTMOUTQ field in the VMBLOK is initialized to the amount of problem state time that the virtual machine is allowed to accumulate before being dropped from a queue. This initial value is set by the scheduler (DMKSCH) when the virtual machine is added to a queue and its value depends on the queue entered (interactive or noninteractive) and on the processor model. For example, the initial value of VMTMOUTQ for a user entering Q1 (interactive) on a Model 145 is 300 milliseconds, while for the same user entering Q2 (noninteractive) it is 2 seconds. Each time the user is dispatched, the value in VMTMOUTQ is entered into the processor timer; whenever the user is interrupted, the decremented processor timer is stored into VMTMOUTQ prior to being set from the new VMTTIME. When the problem state time slice has been exhausted; a processor timer interrupt occurs, the VMQSEND flag bit is set in the VMBLOK, and the scheduler drops the user from the queue. At each queue drop, the problem time used in-queue (the difference between VMTMOUTQ and the initial value) is added to the total problem time field (VMVTIME) in the VMBLOK.

Virtual processor timer simulation is handled for EC mode virtual machines if the value in the virtual processor timer is less than that in VMTMOUTQ. In this case, the VMBLOK is flagged as "tracking processor timer" and a processor timer interrupt is interpreted as a virtual timer interrupt rather than as an in-queue time slice end.

Virtual Timing Facilities

Virtual location X'50' timers are updated by the elapsed processor time each time the dispatcher has been entered after a running user has been interrupted. The size of the update is the difference between the value of the timer at dispatch (saved in QUANTUM at location X'54') and the value of the timer at the time of the interruption (saved in QUANTUMR at location X'4C').

Virtual clock comparator requests are handled by the virtual timer maintenance routine, DMKTMR. They are inserted into the general comparator request stack and the virtual machine is posted when the interruption occurs.

Virtual clock comparator requests to set the virtual processor timer place the new value into the ECBLOK. Requests to store the new value update the ECBLOK field with the virtual processor time used since the last entry to dispatch and pass the value to the user. Requests to set the time-of-day clock are ignored.

A real interval timer or processor timer is one that runs when the virtual machine is executing or is in a self-imposed wait state (that

is, the wait bit is on in the virtual PSW). A real timer does not run if the virtual machine is in a CP pseudo wait state (for example, page wait or I/O wait) or if the virtual machine can be run but is not being dispatched because of other user interaction. Real timers provide accurate interrupts to programs that depend on measurement of elapsed processor and/or wait time. They do not accurately measure wall time -- the TOD clock must be used for this function.

An EC mode virtual machine with the real timer option has both a real interval timer and a real processor timer. Real timer requests for waiting machines are maintained in the clock comparator stack. processor timer requests are added to TOD clock value at the time that they are issued. Interval timer requests must have their units converted. In addition, if the virtual processor timer contains a large negative value, then a real timer request is scheduled to occur when the virtual machine becomes positive, so that the pending timer interruption can be unflagged. Comparator requests for real timer interruptions are inserted into the stack whenever a virtual machine enters a self-imposed wait. They are removed either when the virtual machine resumes execution or when it is forced (or places itself) into a pseudo wait.

I/O Management

I/O SUPERVISOR

The module, DMKIOS, handles the I/O requirements of all system devices except the following terminals: 1052, 3210, 3215, 2150, 2741, 3270 remote equipment, and compatible teletypewriter devices. Scheduling and interruption handling for these devices is essentially a synchronous process and does not require the queuing and restart services of DMKIOS. This is handled by the module DMKCNS. For handling the I/O requirements of 3270 remote equipment, refer to "Programming for 3270 Remote Terminals - an Introduction" in this section.

REAL I/O CONTROL BLOCKS

To schedule I/O requests and control the activity of the I/O devices of the system, I/O control uses several types of control blocks. These blocks are separated into two basic types.

- Static blocks that describe the components of the I/O system.
- The dynamic blocks that represent active and pending requests for I/O operations.

The I/O devices of the real system are described by one control block for each channel, control unit, and device available to the control program. Units present but not represented by control blocks are not available for either user-initiated or CP-initiated operations.

Because all virtual machines are run in the problem state, any attempt to issue a SIO instruction results in a program interruption that indicates a privileged operation exception. This interruption is handled by CP's first level program interrupt handler, DMKPRGIN. It determines if the virtual machine was in virtual supervisor state (problem state bit in the virtual PSW is zero). If so, the instruction causing the interruption is saved in the VMBLOK for the virtual machine and control is transferred to the privileged instruction simulator, DMKPRVLG, via a GOTO.

DMKPRVLG determines if the privileged operation affects the virtual I/O configuration. DMKPRVLG simulates non-I/O privileged instructions (such as LPSW). If the instruction's operation code is from X'9C to X'9F', control is transferred to DMKVSIEX.

After clearing the condition code in the user's VMBLOK, DMKSCNVU is then called to locate the virtual I/O blocks representing the I/O components (channel, control unit and device) addressed by the instruction. DMKVSIEX then branches to handle the request based on the operation requested.

In attached processor systems, the I/O control blocks are protected by forcing all critical execution paths in CP to operate on the main processor.

VIRTUAL I/O REQUESTS

The virtual I/O interface maintained by CP provides to the software operating in the user's virtual machine, the condition codes, CSW status information, and interruptions necessary to make it appear to the user's virtual machine that it is in fact running on a real System/370. The virtual I/O interface consists of:

- A virtual I/O configuration for each active virtual machine that consists of a set of I/O control blocks that are maintained in the Control Program's free storage. This configuration is built at logon time from information contained in the user's directory file, and can be changed by the user or the system operator.
- A set of routines that maintain the status of the virtual I/O configuration.
- Other system routines that simulate or translate the channel programs provided by the user to initiate I/O on units in the real system's configuration.

Virtual SIO

With a SIO, the condition code returned from DMKSCNVU is tested to verify that all addressed components were located. If they were not, then a condition code of 3 (unit not available) is placed in the PSW and control returns to the dispatcher. Otherwise, the addresses of the appropriate virtual I/O control blocks are saved, and DMKVSIEX tests the status of the addressed I/O units by scanning the VCHBLOKs, VCUBLOKS, and VDEVBLOKs to locate the block that contains the status of the addressed subchannel. The subchannel status is indicated in:

- The VCHBLOK for a selector or block multiplexer channel.
- The VCUBLOK for a shared selector subchannel on a byte multiplexer channel.
- The VDEVBLOK for a nonshared subchannel on a byte multiplexer channel.

When the block containing the status is found, the status is tested. If the subchannel is busy or has an interruption pending, condition code 2 is placed in the virtual PSW. Otherwise, the subchannel is available and the device and the control unit are tested for interruption pending

or busy. If either is found, condition code 1 is placed in the virtual PSW and the proper CSW status is stored in the virtual machine's page zero. If all components in the subchannel path are free, DMKVSIEX proceeds to simulate the SIO by locating and loading the contents of the virtual machine's CAW from virtual location X'48' and testing the device type of the unit addressed.

The device type is in the VDEVBLOK. If the device class code indicates a terminal or console, control is passed to the module DMKVCNEX with a GOTO. DMKVCNEX interprets and simulates the entire channel program, moving the necessary data to or from virtual storage and reflecting the proper interruptions and status bytes. When DMKVCNEX has finished, it passes control directly to the dispatcher, DMKDSPCH.

If the referenced device is a spooled unit record device, DMKVSIEX passes control to DMKVSPEX for additional processing. When control returns to DMKVSIEX, it passes control to DMKDSPCH.

If the device is not a terminal or a spooling device, the SIO is translated and executed directly on the real system's I/O device. DMKVSIEX calls DMKFREE to obtain free storage and then it constructs an IOBLOK in the storage obtained. The IOBLOK serves as an identifier of the I/O task to be performed. It contains a pointer to the channel program to be executed and the address of the routine that is to handle any interruptions associated with the operation.

DMKVSIEX stores the contents of the user's CAW in IOBCAW and sets the interruption return address (IOBIRA) to be the same as the virtual interruption return address (DMKVIOIN) in DMKVIO. The CCW translation routine (DMKCCWTR) is then called to locate and bring into real main storage all user pages associated with the channel program, including those containing data and CCWs. The following occurs:

- The CCWs are translated.
- A corresponding real channel program is constructed.
- The data pages are locked into real storage.
- DMKCCWTR returns control to DMKVSIEX. DMKVSIEX places the user in a pseudo wait state, IOWAIT, and calls the real I/O scheduler DMKIOSQV to schedule the I/O on the real configuration.

DMKIOSQV queues the request for operation on the real channel, control unit, and device corresponding to the address used by the virtual machine. When the real SIO is issued, DMKIOS takes the user out of IOWAIT and reflects the condition code for the SIO if it is zero. If it is not zero, the operation is further analyzed by DMKVIOIN. In any case, DMKIOSQV returns control to DMKVSIEX, which passes control to DMKDSPCH.

Other Privileged I/O Instructions

Other privileged I/O instructions are handled directly by DMKVSIEX. DMKVSIEX scans the virtual channel, control unit, and device blocks in the same manner as for a SIO and reflects the proper status and condition to the virtual machine. In some cases (TIO), the status of the addressed devices is altered after the status is presented.

If the operation active on the virtual device is actually in progress in the real equipment, the simulation of a HIO or HDV is somewhat more involved, since it requires the actual execution of the instruction. In this case, the active operation is halted and the resultant condition code/status is returned to the user.

Virtual Channel-to-Channel Adapter

The virtual channel-to-channel adapter (CTCA) simulates data transfer and control communication between two selector channels, either on two distinct processors or two channels on a single processor. Data transfer is accomplished via synchronized complementary I/O commands (for example, read/write, write/read) issued to both parts of the CTCA. Each part of the CTCA is identical and the operation of the unit is completely symmetrical. The CTCA occupies an entire control unit slot on each of the two channels attached. The rightmost four bits of the unit address (device address) are ignored completely and are not available for use.

The VM/370 control program support for virtual CTCA includes all status, sense data, and interruption logic necessary to simulate the operation of the real CTCA. Data transfer, command byte exchange, sense data, and status data presentation for the virtual CTCA is accomplished via storage-to-storage operations (MVCL, etc.). No real I/O operations (excluding paging I/O) nor I/O interruptions are involved. Unit errors or control errors cannot occur.

Virtual Selector Channel I/O Requests

The CCW translator, DMKCCWTR, is called by the virtual machine I/O executive program (DMKVSIEX) when an I/O task block has been created and a list of virtual CCWs associated with a user's SIO request must be translated into real CCWs.

When the I/O operation from a self-modifying channel program is completed, DMKUNTIS is called by DMKIOS. When retranslation of OS ISAM CCWs is required, the self-modifying channel program checking portion of DMKCCWTR calls DMKISMTR.

DMKCCWTR operates in two phases:

- A scan and a translate phase.
- A TIC-scan phase.

A self-modifying channel program checking function is also included.

The scan and translate phase analyzes the virtual CCW list. Some channel commands require additional doublewords for control information (for example, seek addresses). Additional control words are also allocated (in pairs) if the data area specified by a virtual CCW crosses 4096-byte page boundaries, or if the virtual CCW includes an IDA (indirect data address) flag.

Space is obtained from DMKFREE for the real CCW list, and the translation phase then translates the virtual CCW list into a real CCW list. TIC commands that cannot be immediately translated are flagged for later processing by the TIC-scan phase. A READ or WRITE command that

specifies that data cross 4096-byte boundaries is revised to include an IDA flag that points to an indirect data address list (IDAL) and a pair of words for each 4096-byte page, in which each word handles a data transfer of 2048 bytes (or less). The real CCW is flagged as having a CP-generated IDA. DMKPTRAN is called (via the TRANS macro) to lock each 4096-byte page.

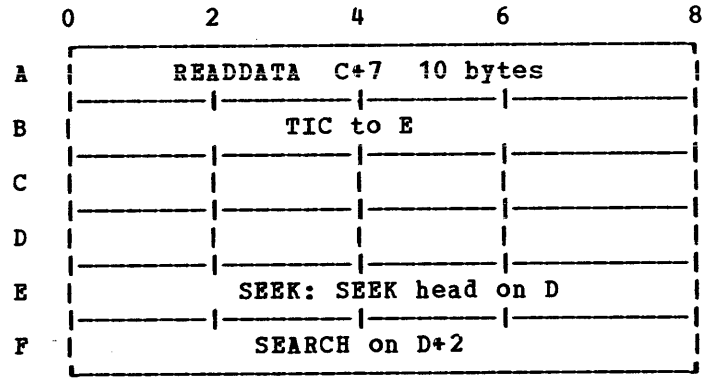
If the real CCW string does not fit in the allocated free storage block, a new block is obtained. The old block is transferred and adjusted before being released. The translation continues with the new block. The process is repeated, as needed, to contain the real CCW string.

Virtual CCWs having an IDA flag set are converted to user translated addresses for each IDAW (indirect data address word) in the virtual IDAL. DMKPTRAN is called for each IDAW is. The CCW is flagged as having a user (but not CP) generated IDA.

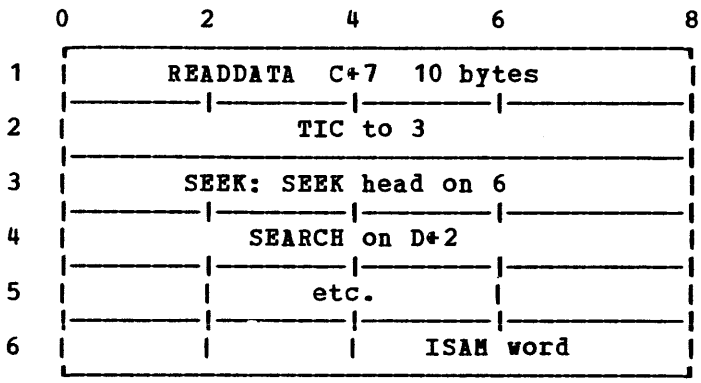
The TIC-scan phase scans the real CCW list for flagged (untranslated) TIC commands and creates a new virtual CCW list for the untranslated commands. Scan-translate phase processing is then repeated. When all virtual CCWs are translated, the virtual CAW in the IOBLOK task block is replaced by the real CAW (that is, a pointer to the real CCW list created by DMKCCWTR), and DMKCCWTR returns control to DMKVIOEX. The user protection key is saved.

OS ISAM Handling by DMKISMTR

Because many of the OS PCP, MFT, and MVT ISAM channel programs are self-modifying, special handling is required by the VM/370 control program to allow virtual machines to use this access method. The particular CCWs that require special handling have the following general format:



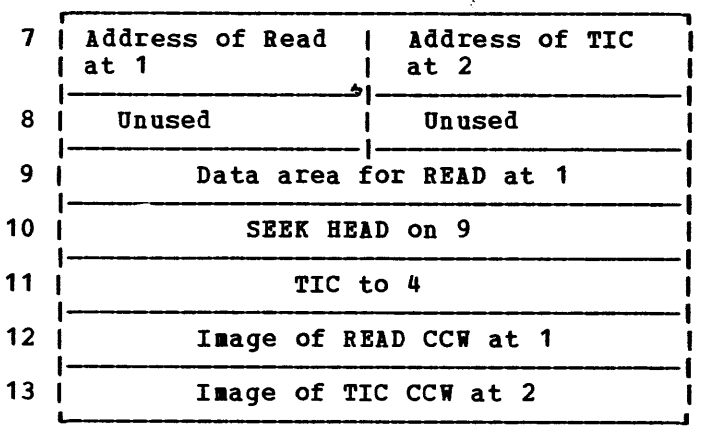
The CCW at A reads 10 bytes of data. The tenth byte forms the command code of the CCW at E. In addition, the data read in makes up the seek and search arguments for the CCWs at E and F. After the CCW string is translated by the VM/370 control program, it usually is in the following format:



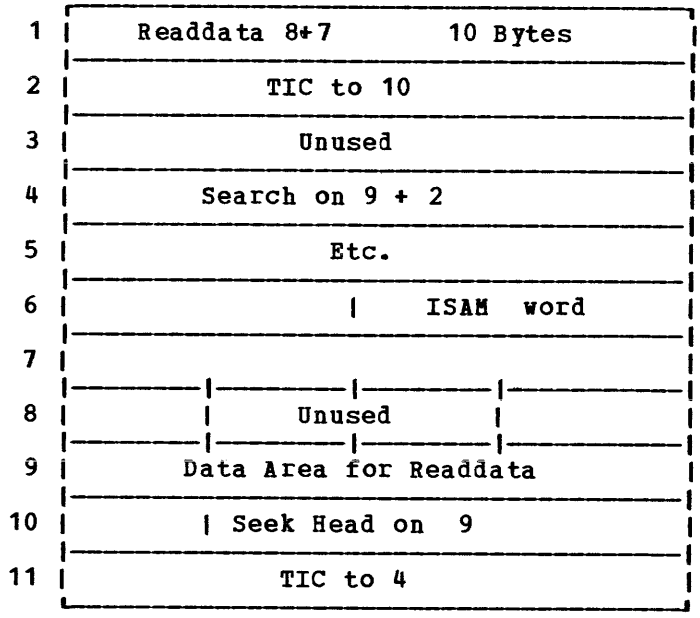
To accomplish an efficient and non-timing-dependent translated operation for OS ISAM, the virtual CCW string is modified in the following manner.

DMKISMTR is called by DMKCCWTR if, during normal translation, a CCW of the type at 1 is encountered. The scan program locates the TIC at 2 by searching the translated CCW strings. The TIC at 2 locates the SEEK at 3.

The virtual address of the virtual SEEK CCW at E is located from the RCWTASK header. Seven doublewords of free storage are obtained and the address of the block is saved in the ISAM control word at 5. The seven doublewords are used to save the following information from the translated CCW strings:



The translated read CCW (at 1) is moved to the save block at 12. The TIC CCW (at 2) is moved to the save block at 13, and the addresses of 1 and 2 are saved at 7. The read CCW at 1 is modified to point to a 10-byte data area at 8+7 in the save block. The seek head CCW at 3 is copied into the save block at 10, and the seek address is modified to point to the data area at 9. At 11, a TIC CCW is built to rejoin the translated CCW string at 4. The search at 4 (or any subsequent search referencing D+2) is modified to point to 9+2. The completed CCW string has the following format:



The interruption return address in the IOBLOK is set to DMKUNTIS. DMKUNTIS restores the CCWs to their original format from the seven doubleword extensions, moves the 10 bytes of data from 8+7 into virtual storage (at C+7), and releases the block. Normal I/O handling is resumed by DMKVIO and DMKUNT.

I/O COMPONENT STATES

The I/O components represented by the control blocks described in "Real I/O Control Blocks" are in one of four states and the state is indicated by the flag bits in the block status byte. If the component is not disabled, it is either busy, scheduled, or available.

If the disabled bit is on, the component has been taken offline by the operator or the system and is at least temporarily unavailable. A request to use a disabled component causes the IOBLOK to be stacked with an indication of condition code 3 on the SIO and the real SIO is not performed.

An I/O unit is busy if it is transferring data (in the case of a channel or control unit), or if it is in physical motion (in the case of a device). If an I/O unit is busy, the IOBLOK for the request is queued from the control block representing that I/O unit.

An I/O unit is scheduled if it is not busy but will become busy after a higher-level component in the subchannel path becomes available and an operation is started. For example, if a request is made to read from a tape drive and the drive and control unit are available, but the channel is busy, the IOBLOK for that request is queued from the RCHBLOK for the busy channel and the RCUBLOK and RDEVBLOK of the drive and control unit are marked as scheduled. Future requests to that drive are queued from the RDEVBLOK for the scheduled device. When the channel completes the operation, the next pending operation is dequeued and started; the scheduled control unit and device are then marked as busy.

The IOBLOKs for various I/O requests indicate the status of that request by a combination of the status bits in the IOBLOK and the queue

in which the block resides. In general, an IOBLOK is queued from the control block of the highest level I/O unit (taken from device up to channel) in the subchannel path that is not available. Once the I/O operation is started, the IOBLOK is chained from the active IOBLOK pointer (RDEVAIOB) in the real device control block. Flags in the IOBLOK status fields may also indicate that a unit check has occurred, that a sense is in progress, or that a fatal I/O error (unrecoverable) has been recognized by error recovery procedures. After I/O control releases control of the IOBLOK, it is stacked on the queue of IOBLOKS and CPEXBLOKs anchored at DMKDSPRQ in the dispatcher and control is passed to the second-level interruption handler whose address is stored in IOBIRA.

I/O INTERRUPTIONS

I/O interruptions are either synchronous or asynchronous. Asynchronous interruptions indicate the change in status of an I/O unit from the not-ready to ready state or busy to not-busy state. In either case, if the affected component has any pending requests queued from its control block, they are restarted, and whether or not the given interrupt is processed any further depends upon the status of the interrupting component. Channel-available and control-unit-end interruptions restart the interrupting component. An asynchronous device end is passed to the user if the device is dedicated; otherwise, the device is restarted.

An interruption is considered to be synchronous if the interrupting device has a nonzero pointer to an active IOBLOK. In this case, the following processing occurs:

- If a unit check has occurred, a sense is scheduled, and when the sense is completed, the appropriate ERP is called.
- If an ERP is currently in control of the task (indicated by a flag in the IOBLOK), return the IOBLOK to the appropriate ERP.
- If the operation is incomplete (for example, channel end is received without device end), the IOBLOK is copied and the copy is stacked but the original IOBLOK remains attached to RDEVAIOB to receive the final interrupt; then, the control unit and the channel is restarted.
- If the operation is complete (that is, the device is available), the IOBLOK is detached from the device and stacked, and the device, control unit and channel are restarted.

The restart operation usually dequeues the next IOBLOK that is queued to the restarted component and queues it to the next higher component in the subchannel path. When the channel level is reached, a SIO is issued and exit is taken to the dispatcher after handling any nonzero condition codes as previously described.

VIRTUAL I/O INTERRUPTIONS

When an I/O interruption is received, the IOBLOK is stacked for dispatching and control is passed to the address specified in the IOBIRA (interrupt return address) field. For operations requested by DMKVIOEX, the return address is DMKVIOIN (virtual interrupt return address). When DMKVIOIN receives control from the dispatcher, it loads the virtual address of the unit with which the interruption is associated from the IOBLOK and calls DMKSCNVU to locate the virtual device control blocks.

DMKVIOIN then tests the IOBLOK status field to determine the cause for the interruption. If the block has been unstacked because of an interruption, the field is zero. If the operation was not started, it contains the condition code from the real SIO.

Note: The VIRA should not see a real condition code 2 as the result of a SIO, since channel-busy conditions are detected and reflected before any real I/O operation is attempted.

A condition code of 3 is reflected virtual machine and exit is taken to the to the dispatcher. For a condition code of 1, the CSW status field in the IOBLOK is examined to determine the cause for the CSW stored condition. The status is reflected to the virtual machine and various components of the virtual configuration may be freed, if the status so indicates. For example, if the CSW status indicated both channel end and device end, the operation was immediate and has completed. Thus, the CCW string (real) may be released and all virtual components marked available.

The CSW status returned for a virtual interruption must be tested in the same manner, with the additional requirement that the status be saved in the affected virtual I/O control blocks and that the CSW be saved in the VDEVCSW field for the device causing the interruption. If the unit check bit is on in the status field, the sense information saved in the associated IOERBLOK (pointed to by the IOBLOK) must be retained so that a sense initiated by the virtual machine receives the proper information.

In any case, when an interruption is received for a virtual device, a bit in the interruption mask, VCUDVINT, for the device's control unit is set to 1. The bit that is set is the one corresponding to the relative address of the interrupting device on the control unit. For example, if device 235 interrupts, the fifth bit in the VCUDVINT mask in the VCUBLOK for control unit 30 on channel 2 is flagged. Similarly, the bit in the VCHCUINT in the affected VCHBLOK is also set; in this case, bit 3 in VCHBLOK for channel 2. If the interruption is a channel class interrupt (PCI or CE), the address of the interrupting unit (235) is stored in the VCHCEDEV field in the VCHBLOK. The final interruption flag is set in the VMPEND field in the VMBLOK for the interrupted virtual machine; the bit set corresponds to the address of the interrupting channel. The next time, the virtual machine is dispatched and becomes enabled for I/O.

SCHEDULING I/O REQUESTS

A task that requests an I/O operation must specify the device on which the operation is to take place and must provide an IOBLOK that describes the operation. Upon entry to DMKIOS, register 10 must point to the IOBLOK. The IOBLOK must contain at least a pointer to the channel program to be started in IOBCAW and the address to which the dispatcher is to pass control in IOBIRA. In addition, the flags and status fields should be set to zero. If the operation is a VM/370 control program function such as for spooling or paging, the entry point DMKIOSQR is called. If the requester is the virtual I/O executive (DMKVIOEX) attempting to start a virtual machine operation, the entry point DMKIOSQV is called and some additional housekeeping is done. In either case, an attempt is made to find an available subchannel path from the device to its control unit and channel. If an I/O unit in the path is busy or scheduled, the IOBLOK for the request is queued to the control block of the I/O unit.

Requests are usually queued first-in-first-out (FIFO), except those requests:

- To movable-head DASDs that are queued in order of seek address
- That release the affected component after initiation (SEEKS and other control commands) which are queued last-in-first-out (LIFO) from the control block

Whether or not the operation has been successfully started, the caller requesting the I/O operation receives control from DMKIOS. If a free path to the device is found, the unit address is constructed and an SIO is issued. If the resulting condition code is zero, control is returned to the caller; otherwise, the code is stored in the requester's IOBLOK along with any pertinent CSW status, the IOBLOK is stacked, any components that become available are restarted, and control is returned to the caller.

Alternate Path Scheduling

Alternate path I/O scheduling is performed according to the following scheme:

DMKIOS searches for an available path beginning with the primary path to the device. If an available path to the device exists, the I/O request is started immediately on the first available path to the device.

If the device is busy or scheduled, the IOBLOK is queued off the RDEVBLOK. No alternate path processing is performed at the device level.

If the device is not busy, not scheduled, nor offline, an IOBLOK for this I/O request is promoted upward to the RCUBLOK or RCHBLOK level in search of an available path. If a busy or scheduled path is encountered, an IOBLOK is queued to the real block and the search continues for an available path. If more than one busy path is encountered, multiple IOBLOKs are queued for the same I/O request. This is accomplished by creating mini IOBLOKs for each busy/scheduled path after the first. The primary IOBLOK is queued off the first busy path encountered. The mini IOBLOK is 16 bytes in length and consists of the first two doublewords of the IOBLOK, which is the same as the current IOBLOK structure. The IOBLOK and associated mini IOBLOKs are chained in a single-threaded queue by means of the IOBLINK field. The active IOBLOK pointer is not stored in the IOBLINK field until just prior to the SIO. Zeros are stored in IOBLINK at entry to DMKIOSQR to indicate no mini IOBLOKs have been queued as yet. See Figure 17 for an example of mini IOBLOK queuing.

The last two words of the mini IOBLOK (IOBFPNT and IOBBPNT) are used as the double-threaded queue pointers for the RCUBLOK/RCHBLOK from which it is queued. A flag is set in the mini IOBLOK to identify it as a mini IOBLOK.

Figure 18 shows a sample control block structure when mini IOBLOKs are queued.

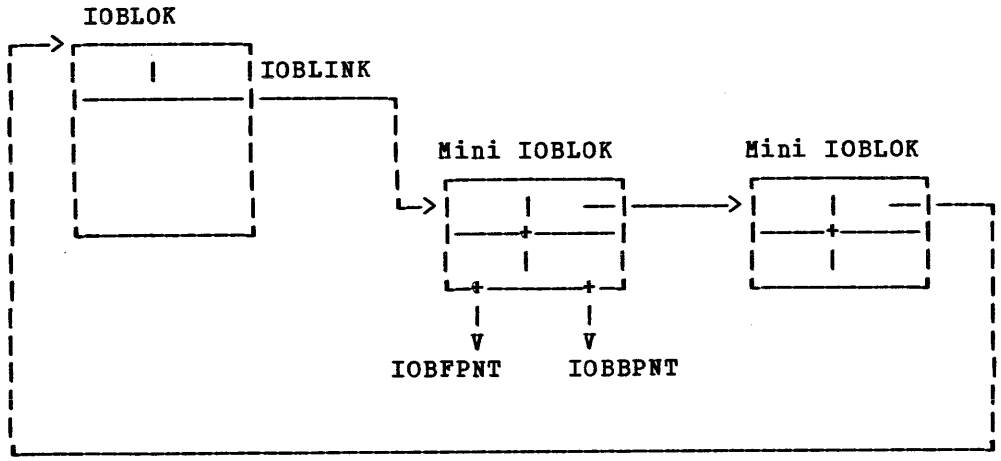


Figure 17. Mini IOBLOK Queuing

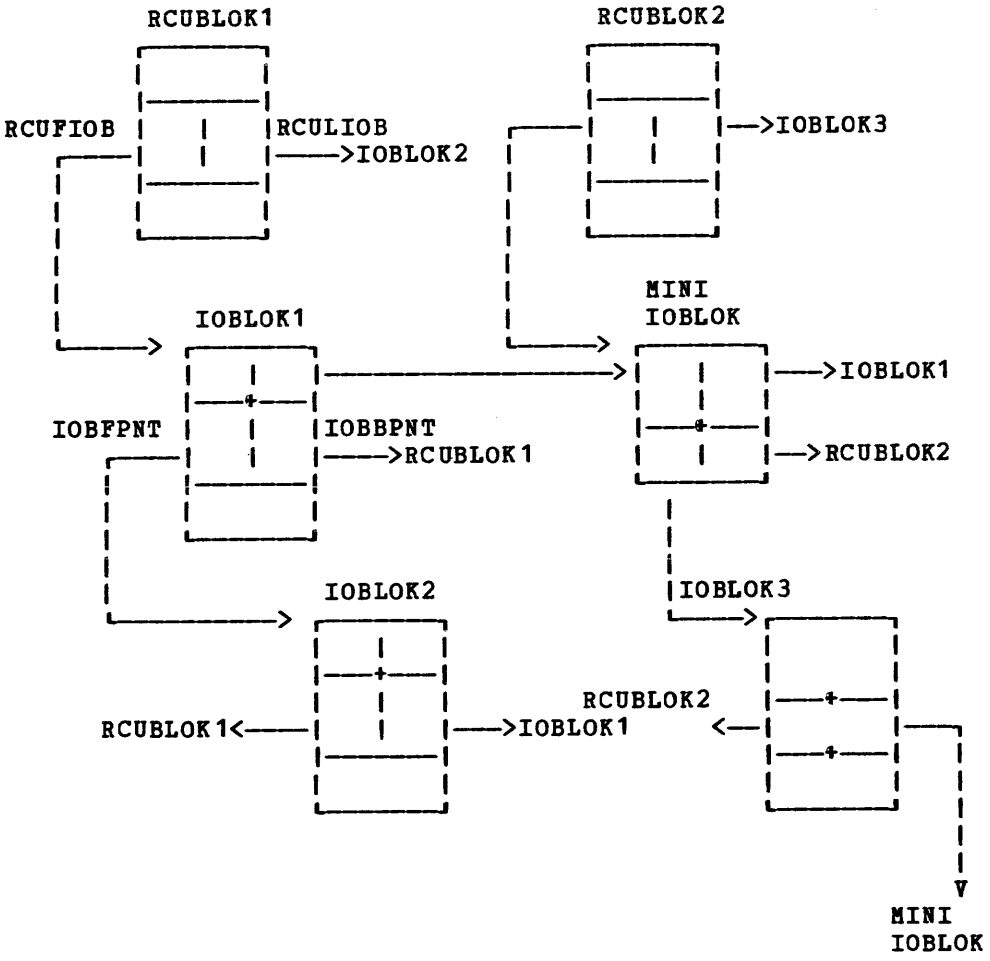


Figure 18. Control Block Structure for Alternate Path Request

Prior to starting an I/O operation associated with the request, a check is made to see if the IOBLOK is a mini IOBLOK and whether mini IOBLOKs are queued off this IOBLOK. All mini IOBLOKs associated with this request are dequeued from their respective queues by running the IOBLINK chain. The storage for the blocks is released. If the active IOBLOK is a mini IOBLOK, the IOBRADD from the mini IOBLOK is moved to the primary IOBLOK and the I/O started using the primary IOBLOK. Once the storage for the mini IOBLOKs has been released and the SIO initiated, any busy condition encountered causes the IOBLOK to be queued on this same path. That is, there will be no alternate path scheduling after the SIO if a busy condition is encountered. The I/O request will be restarted on the same path as the original request.

Reserve/Release

Reserve/release is supported for shared DASD as though each virtual machine has a separate channel path to a shared device. Reserve/release support prevents the occurrence of a channel lockout situation. This is accomplished by changing reserve CCWs to sense CCWs when a reserve is issued to a device that has alternate paths defined to it. This means that whenever alternate paths are defined to a device, the real reserve does not execute on the hardware. Reserve/release support is implemented in VM/370 on a virtual basis allowing the reserve/release operation codes to be simulated on a virtual basis for minidisks, including full-extent minidisks. When a reserve is issued against a minidisk, the reserve is accomplished by a locking mechanism. The status of the minidisk is maintained in the VRRBLOK that is chained from the VDEVBLOK.

The following matrix identifies how the reserve operation code is handled in the various situations.

	Defined Alternate Paths to Device	Will Reserve/Release Execute on the Hardware	Virtual Reserve/Release Requested for Minidisks	RESERVE[1] — or — SENSE[2]
Dedicated DASD or Tape	NO	N/A	N/A	RESERVE
	YES	N/A	N/A	SENSE
Minidisk	NO	NO	NO	RESERVE
	NO	NO	YES	SENSE
	NO	YES	NO	RESERVE
	NO	YES	YES	RESERVE
	YES	N/A	N/A	SENSE

[1]The 'RESERVE' keyword in the chart indicates that the real reserve is allowed to execute on the hardware.

[2]The SENSE keyword indicates that the reserve CCW is changed to a sense CCW. Virtual Reserve/Release is requested by means of a new option on the MDISK directory control statement.

DMKVIO

DMKVIO performs the following steps when virtual reserve/release processing is requested:

1. DMKVSI calls DMKCCW to perform CCW translation. For DASD devices, DMKCCW checks if the virtual reserve/release feature bit is on in the VDEVBLOK. If virtual reserve/release processing has been requested and if the device is not reserved by anyone or it is reserved by this user, processing continues normally. If the device is reserved by another user, DMKCCW calls DMKUNTFR to restore the CCWs to their original state and returns to the caller, unless sense bytes have been transferred to the user's storage in which case CP enqueues on the minidisk and waits until it is no longer reserved at which time the I/O can proceed. If the I/O request can continue and the CCW chain contains a reserve command, the VDEVBLOK and the VRRBLOK are flagged as reserved. If the CCW chain also contains a release, the IOBLOK is flagged to indicate to DMKUNTFR to release the virtual disk. Control returns to DMKVSI.

2. DMKVSI reflects a device-busy condition to the virtual machine if the minidisk is currently reserved by another user.

3. DMKUNT reflects a device end interrupt to all virtual machine users who previously received a busy condition, when the device is released.

Ordered Seek Queuing: Requests to start I/O on system devices are normally handled first in first out. However, requests to movable-head DASD devices are queued on the device in ascending order by seek address. This ordered seek queuing is performed to minimize intercylinder seek times and to improve the overall throughput of the I/O system.

CP assumes that very few virtual machines perform chained SEEKs. Therefore, the first logical address represents the position of the arm upon completion of the I/O operation. Ordered SEEK queuing is based on the relocated real cylinder. DMKIOS uses the cylinder location supplied in IOBCYL for ordered SEEK queuing. This field is initialized by the calling CP routine for paging and spooling or by the CCW translator for virtual I/O. The CCW translator, DMKCCW, supplies the IOBCYL value in the following manner:

- Reads the IPL record, relocates to virtual cylinder 0
- Recalibrates, issues a real calibrate, and then a SEEK to virtual cylinder 0
- Issues a channel SEEK, relocates to the virtual cylinder

The IOBLOK queuing subroutine of DMKIOS recognizes that a request is being queued on a movable-head DASD by means of the device class and type fields of RDEVBLOK. Instead of adding the IOBLOK to the end of the queue on the RDEVBLOK, the queuing routine sorts the block into the queue based on the cylinder number for the request. The cylinder number for any request to DASD is recorded in the IOBCYL field. The queue of IOBLOKs on a real device block is sorted in ascending order by SEEK address, unless the entire device is dedicated to a given user. In this case, DMKIOS does not automatically schedule the device, and no more than one request can be outstanding at any one time.

When an outstanding I/O request for a device has completed, DMKIOS attempts to restart the device by dequeuing and starting the next IOBLOK queued on the device. For non-DASD, this is the first IOBLOK queued. However, for movable-head DASD, the queued requests are dequeued in either ascending or descending order, depending upon the current position (recorded in RDEVCYL) and the direction of motion of the arm. If the arm is seeking up (that is, toward the higher cylinder numbers), the queue of IOBLOKs is scanned from the first block toward the last until an IOBLOK is found with an IOBCYL value equal to or greater than the value in RDEVCYL, or until the end of the queue is reached. At this point, the device is flagged as seeking down and the queue is scanned from last to first until an IOBLOK with an IOBCYL value equal to or less than RDEVCYL is found. When the IOBLOK is found, it is dequeued and started. The direction of motion is indicated by an RDEVFLAG bit and the next request is dequeued downward until the head of the queue is reached.

Because the queue itself is a two-way chained list, no special handling for null or unity set lists is required, and the ordered seek algorithm returns to first-in-first-out queuing.

Dedicated Channel Support: One of the facilities of the VM/370 control program allows a virtual machine to control one or more channels on a dedicated basis. The channels are attached to the virtual machine by using the privileged ATTACH CHANNEL command. A virtual machine can have one or more dedicated channels. In addition, channels can be split between virtual machines but a dedicated channel cannot be shared between two virtual machines. For instance, channel 1 could be dedicated to virtual machine A, and channel 2 could be dedicated to virtual machine B, or both could be dedicated to virtual machine A or B.

With a dedicated channel, all virtual machine device addresses must be identical to the real machine device addresses. For instance, virtual device 130 must be real device 130, and virtual device 132 must be real device 132. With dedicated channels, CP does not perform any virtual device address mapping.

CP error recording and channel recovery procedures are still in effect for dedicated channels. The dedicated channel support can be used in conjunction with the virtual=real feature for any virtual machine that is occupying the virtual=real storage space.

VIRTUAL CONSOLE SIMULATION

DMKVCN receives control from the virtual machine I/O executive, DMKVIO. When control is received, the device is available with no interruptions pending. A console control block, VCONCTL, that is obtained from storage and chained from the virtual device control block, VDEVBLOCK, by DMKLOG is accessed for use during the interpretation of the virtual console I/O sequence. The user's CAW is examined for validity. If it is valid, the TRANS macro is issued to fetch the first user CCW. This CCW is moved to the VCONCTL block for analysis.

The CCW is analyzed to determine if it is a read, a write, a control, a sense, a TIC, or an invalid operation. Based upon the analysis, the appropriate processing routine in DMKVCN is invoked.

The Read Simulation Routine: Obtains a buffer for input data from free storage. The location of the buffer is set in the VCONCTL block. The DMKQCNRD routine is called to schedule and perform an actual read to the corresponding real device representing the user's virtual console. If

SET LINEDIT ON is specified, the buffer data is edited and translated to EBCDIC. When the read is completed, the data is moved to the specified user address obtained from the address portion of the virtual CCW. If command chaining is specified, processing returns to fetch and analyze the next CCW. If command chaining is not specified, the virtual CSW is constructed in the VDEVBLOK and an interrupt is flagged as pending in the VMBLOK.

The Write Simulation Routine: Obtains a buffer for the construction of the output message from free storage. The virtual machine data is located from the virtual CCW address in the VCONCTL block and moved to the data buffer. The DMKQCNWT routine is called to write the data in the buffer and provide the necessary length, translation, and format functions. Control is received at the DMKVCN module upon completion of the writing. At this point, the virtual CCW is re-examined. If command chaining is specified, processing continues to fetch and analyze the next CCW. If command chaining is not specified, the virtual CSW is constructed in the VDEVBLOK and an interruption is flagged as pending in the VMBLOK.

The Control Simulation Routine: Is used for the NOP and ALARM operations. A NOP operation requires no data transfer or I/O operation. An ALARM operation has no equivalent on low-speed teleprocessing equipment; thus, a message indicating the ALARM operation is constructed. DMKQCNWT is called to output the constructed message. If the command is chained, processing continues (for NOP or ALARM) to fetch the next CCW and analyze it. If command chaining is not specified and this is not the first CCW, a virtual CSW is constructed in the VDEVBLOK and an interruption is flagged as pending in the VMBLOK. If this is the first (and only) CCW, then a condition code of 1 is presented with channel end and device end in the virtual CSW.

A Virtual Sense Operation: Is similar to a control operation, because no actual I/O operation is performed. However, there is data transfer. The sense data from the VDEVBLOK is moved to the virtual storage location specified in the virtual CCW address. If the command is chained, processing continues to fetch the next CCW and analyze it. Otherwise, an interruption is flagged as pending in the VMBLOK.

A Virtual TIC Operation: Fetches the virtual CCW addressed by the TIC address and analyzes the fetched CCW. If the fetched CCW is itself a TIC, or if the TIC is the first CCW, a channel program check condition is reflected to the virtual machine as an interruption or as a CSW-stored condition, respectively.

Invalid Operation: Any other operation is considered invalid. Command reject status is posted in the virtual sense byte and the operation is terminated with unit check status presented in the virtual CSW.

REMOTE 3270 PROGRAMMING

For a basic understanding of CP processing of data relating to 3270 devices on binary synchronous lines, the information and terminology contained in IBM 3270 Information Display System Component Description, and General Information - Binary Synchronous Communications is required.

A digest of some of this essential information as it applies to VM/370 follows:

- Text messages to and from remote terminals and printers can only be achieved when the bisync line is in text mode.

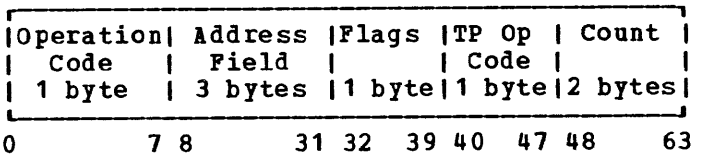
- Text messages from a remote device can be the result of a general poll or specific poll operation to the related device or devices on the bisync line. This polling communication interface is accomplished by each line-connected control unit having unique specific poll and general poll recognition circuitry and by the CP terminal list of valid bisync lines and 3270 remote control unit addresses. This list, the terminal list, is generated by VM/370 system generation procedures employing TERMINAL and CLUSTER macros. For more details about terminal list generation, see the VM/370 Planning and System Generation Guide.

- Reliability and dependability of line operation is achieved by the use of: a double addressing scheme, control characters with a rigid message protocol, and complex redundancy-check characters appended to transmission messages. Examples of these techniques are shown in the formats that follow.

- Every message (text or control) that is issued by CP may or may not be responded to by the remote station or control unit. The type of response (or absence of response) that CP receives depends on the receptiveness of that device or control unit to the previously sent message (is the device ready and enabled and accurately addressed) and the content and correctness of the message (no line errors).

- To establish the relationship of the line of terminal response to a particular line or device write or read operation, CP employs an operation "tracking" facility (TP op code) imbedded in the issued CCWs. The function performed by the CP op code is described in the following CCW formats.

Format of the 3270 Remote CCW



where:

Operation Code
contains the hexadecimal value of the type of operation performed by the command.

Valid operation codes are:

X'01' WRITE
X'02' READ
X'03' NO-OP
X'09' POLL
X'23' SET MODE
X'27' ENABLE
X'2F' DISABLE

Address Field
Depending on CCW usage, this field may address an:

Area
: The address of the data area (read buffer) located in the BSCBLOK at BSCREAD.

Table
: The appropriate location in the table of data-link control characters provided in the module DMKGRF (Example: RVI, EOT, ENQ).

Response
: (BSCRESP). The address location of the response message in the BSCBLOK.

List
: The appropriate entry in terminal list (NICBLOKS) associated with the READ or WRITE operation. The entry for WRITE operation is at location BSCSEL. The entry for the READ operation is at location BSCPOLL.

Note: To see how the key words AREA, TABLE, RESPONSE, and LIST are used, refer to the CCW sequences described in "I/O Program Routines for Bisync Lines and 3270 Remote Devices" in this section.

Flags
: The flag bits turned on in the CCW: CC (channel commands), CD (chained data), SILI (suppress incorrect length indication), skip (suppress data transfer to main storage) and PCI (program-controlled interrupt).

TP Op Code
: An imbedded teleprocessing operation code in the CCWs used in bisync line communications. This code is inspected by the secondary interruption handler, DMKRGAIN, when channel end and device end are received. The code is also used by the error processing module, DMKBSC. The code indicates the function being performed by the associated command. For use of the TP op codes, refer to the formatted CCWs that follow.

Count
: Refers to the byte length of the CCW READ or WRITE operation.

I/O PROGRAMS FOR BISYNCHRONOUS LINES AND REMOTE 3270S

Before data communication to remote 3270 equipment can take place, the remote teleprocessing line, the control unit and the device(s) must be enabled for communication. This occurs when control unit hardware recognizes a unique string of characters transmitted on the line from CP. Disabling a line occurs in a similar manner. The following is the format of the CCWs used in the enabling/disabling operation:

Enable a Line

Operation	Command Code	Address	Flags	TP Op Code	Count
Disable Line	X'2F'	0	CC, SILI	01	1
Set Mode	X'23'	X'40'	CC, SILI	01	1
Enable Line	X'27'	0	SILI	01	1

Disable a Line

Operation	Command Code	Address	Flags	TP Op Code	Count
Disable Line	X'2F'	0	SILI	01	1

After a line is enabled, communication can then be directed to a particular resource. The sequence of events (for a write disable and write continue) is as follows:

Send a data link control character on the line that places the control unit in control mode. This mode makes the control unit receptive to the specific address indicated by the second CCW. The third CCW is a read CCW that is needed for the acknowledgement response from the addressed control unit. Normally, in response, CP transmits a block of data to that device with a write text CCW. Acknowledgement of receipt of this data is contained by the read response (write continue) CCW. The format of the CCW write initial and write continue operation follows.

Write Initial

Operation	Command Code	Address	Flags	TP Op Code	Count
Write an EOT	01	Table	CC, SILI	02	1
Write address-ing char.	01	List	CC, SILI	03	LIST
Read Response	02	Response	SILI	05	2

Write Continue

Operation	Command Code	Address	Flags	TP Op Code	Count
Write text	01	Area	CC, SILI	10	variable
Read Response	02	Response	SILI	11	2

In situations where the line is found to be in text mode, CP can issue a write reset sequence to put the binary synchronous line in control mode. The following format illustrates the write reset CCW.

Write Reset

Operation	Command Code	Address	Flags	TP Op Code	Count
Write EOT	01	Table	SILI	09	1

In situations where the expected response from a remote station was not received or was invalid, the channel program may request the remote station to retransmit the response. The following write ENQ format shows this sequence. The remote station, upon receipt of the ENQ message, responds by transmitting the expected or valid response to the response area indicated by the second CCW.

Write ENQ

Operation	Command Code	Address	Flags	TP Op Code	Count
Write ENQ	01	Table	CC, SILI	03	1
Read Response	02	Response	SILI	11	2

Read operations occur following a general poll or a specific poll for text messages. In a general poll sequence, CP transmits the general poll characters to the attached control unit on the bisync line. The control unit recognizes the polling request, then the list (referred to in the poll CCW) of enabled devices is scanned for any messages that are queued and ready for transmission. A positive acknowledgement (yes, I have a message to transmit) from any of the attached devices causes the next CCW to be skipped. The last CCW provides the read buffer and the count necessary for the incoming data block from the first remote

station on the list that had a message queued for transmission. If, however, all remote stations respond with negative acknowledgement (no messages queued) or any station queried for a response fails to respond, then the channel program ends with the third CCW. The following read initial format shows the initial read CCW sequence.

Read Initial

Operation	Command Code	Address	Flags	TP Op Code	Count
Write EOT	01	Table	CC, SILI	02	1
Poll	09	List	CC, SILI	03	LIST
T/O No-operation	03	0	SILI	07	1
Read Text	02	Area	SILI	10	162

After CP receives a message from a remote station, it may reissue the initial read sequence to poll the remaining stations on the list (assuming the list of enabled devices was not exhausted on the first pass of the initial read sequence). In the event that the list was exhausted on either the first or a subsequent initial read sequence, CP starts the poll delay, then allows the poll delay interval to expire before starting another read scan to the line (assuming CP has no higher line priority tasks to process). If, in the process of receiving messages from remote stations, CP receives a message block that is invalid or its beginning or ending bisync control characters are not recognized, CP can elect to send a negative response back to the remote station. This negative response, the NAK control character, causes the remote station to retransmit the previous message to CP; this incoming message is processed by the second CCW of the read repeat sequence as shown in the format below.

Read Repeat

Operation	Command Code	Address	Flags	TP Op Code	Count
Write NAK	01	Table	CC, SILI	06	1
Read Text	02	Area	SILI	10	162

Once CP message processing receives an error-free message from a remote station, CP sends an RVI control character to the remote station before processing the message. The remote station, upon recognition of the RVI character, halts the sending of additional queued data and responds with EOT (instead of the normal ACK0/ACK1 response). The

second CCW of the read interruption sequence processes the EOT response from the remote station as shown in the format below.

Read Interruption

Operation	Command Code	Address	Flags	TP Op Code	Count
Write RVI	X'01'	Table	CC, SILI	06	2
Read Response	X'02'	Response	SILI	11	2

DATA FORMATS - BISYNCHRONOUS LINES AND REMOTE 3270S

CP, in conjunction with remote 3270 support, uses the following formats for its text messages. For a detailed explanation of the abbreviations used, see the IBM 3270 Information Display System Component Description.

Write Text Data Message Format

Display commands use this message format for the placement or erasure of data anywhere on the display screen. The display commands that implement this function are: WRITE (X'F1'), ERASE/WRITE (X'F7') and COPY (X'F7').

Write Data Stream

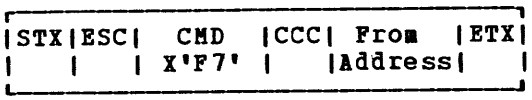
STX	ESC	CMD	WCC	BSA	Buffer Address	Orders & Text	SBA	Buffer Address	ETX
1	1	1	1	1	2	variable	1	2	1

Write Text Messages for the Copy Command

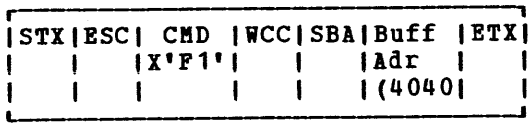
The COPY command is limited to compatible printers located on the same control unit. Action starts by pressing a PF key designated for the COPY function. CP responds by sending a message to the control unit that contains both the designated printer and the display station that requested the action and directs the control unit to print the designated display buffer to the printer specified.

The format of the COPY messages follows:

3271 Copy Data Stream



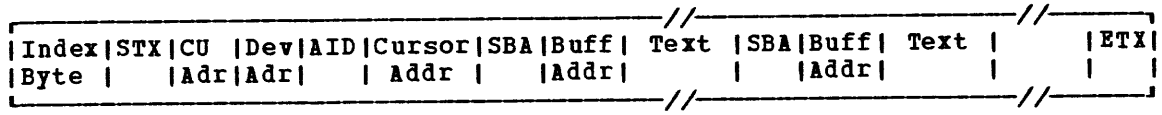
3275 Copy Data Stream



Read Text and Read Header Message Formats

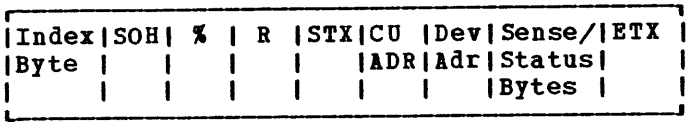
The following is representative of typical input-to-processor message formats. The format of a multiline read operation follows.

Read Text Data Stream



Error Status Data Stream

Another form of input message is the error status message. Error status is processed by the DMKRGF module. The characters, %R, following the SOH signify that this message contains sense and status data. The format of this message follows.



Test Request Data Stream

The test request message, upon receipt from display terminals, is ignored by CP. The input inhibit mode that the display terminal enters upon pressing the test request key can be reset only if the terminal user presses the RESET key. The characters, %/, following SOH indicate the test request function. The format of this message follows.

ALLOCATION MANAGEMENT

Real storage space above the Control Program nucleus is made up of the dynamic paging area and the free storage area. Page frames (allocation space in real storage for a page of data) in the dynamic paging area are allocated to virtual machines and the control program to satisfy paging requests. Blocks of storage, requested by virtual machines and CP for working storage, are allocated from the free storage area.

NORMAL PAGING REQUESTS

If a program interruption is caused by a normal paging request (not from a virtual machine that is running in EC mode with translation on), DMKPRGIN determines whether a segment or page translation error has occurred. If one of these errors occurred, an invalid address interruption code is set, and the interruption is reflected to the virtual machine supervisor. If a segment or page translation error has not occurred, the virtual machine's current PSW is updated from the program old PSW (PROPSW), the address of the current VMBLOK is placed in register 11, and DMKPTRAN is called to obtain the required page. When the paging operation is completed, control is returned to DMKDSPCH.

Virtual Storage Management

When operating in the CP relocate environment, each virtual machine's virtual storage space is described by two sets of tables.

- One set, the segment and page tables, describes the location and availability of any of the virtual machine's virtual pages that may be resident in real storage. Locations in these tables are indexable by virtual address, and the entries contain index values that reference corresponding real storage addresses. In addition, each table entry contains an indication of whether the corresponding virtual page is available to the user in real storage. These tables are referenced directly by the DAT feature when the virtual machine's program is running.

- The second set of tables, called swap tables, is a map of the locations of the virtual machine's pages on the DASD devices that comprise the system's paging or auxiliary storage. The DASD addresses in these tables can either represent the source of a page of virtual storage (the location to which a page may be moved if necessary) or a dummy address, indicating that the given page has not yet been referenced and, thus, has a value of binary zeros.

  The swap tables are arranged in a format indexable by virtual storage address. In addition to containing the address of a page, each entry contains flags and status bytes that indicate such information as:

- The storage protection keys to be assigned to the page when it is made resident.

- Whether the page is currently on its way into or out of the system (in transit), etc.

These tables are not referenced directly by the hardware as are the page and segment tables, but are used by paging management to locate user pages that are needed to execute a program.

Virtual storage management is done by the technique known as demand paging. This means that a page of virtual storage is not "paged in" from its DASD auxiliary storage area until it is needed. CP does not determine the pages required by a virtual machine before the virtual machine executes. A demand for a page can be made either implicitly by the virtual machine or explicitly by CP.

- An implicit demand for a page is made when a program attempts to reference a page that is not available in real main storage. This attempt causes a program interruption with the interruption code indicating a page or segment exception. Upon recognition of this condition, control is passed to the paging manager to obtain a page frame of real main storage and to bring in the desired page.

- An explicit demand for a page can be made by CP (for example, in the course of translating a user's channel program). If, in the process of translation, CP encounters a CCW that addresses a page that is not resident in real storage, a call is made to the paging manager to make the referenced page resident.

While the requested page is being fetched, the requesting virtual machine is unable to continue execution; however, it may be possible to run other tasks in the system, and CP runs these while the needed page is being paged in. When the requested page is resident, the virtual machine can be run and is dispatched in its turn.

In addition to demanding pages, virtual machines implicitly or explicitly release page frames of their virtual storage space. Part of the space may be explicitly released from both real and virtual storage via a DIAGNOSE instruction that indicates to the control program those page frames that are to be released. An entire virtual storage is released when a user loads (via IPL) a new operating system or logs off from the system.

CP also has virtual storage associated with it. This space contains CP (some parts of which need not always be resident in real storage), and virtual storage buffers for spooling and system directory operations. Although CP makes use of virtual storage space for its execution, it does not run in relocate mode. Thus, nonresident modules must be completely relocatable.

Real Storage Management

Real storage management allocates the system's page frames of real storage to satisfy the demands for virtual pages made by the system's virtual machines. Efficiency of allocation involves a trade-off; the paging manager uses only enough processor time to ensure that:

- The set of virtual storage pages that are resident represent those pages that are most likely to be used.

- A sufficient number of cycles is available to execute virtual machine programs.

Inefficiency in the first area causes a condition known as thrashing, which means that frequently used pages are not allowed to remain resident long enough for useful work to be performed by or on them. Thrashing could be aggravated by the paging manager's page frame selection algorithm or by a dispatcher that attempts to run more tasks than the system can handle (the sum of their storage requirements

exceeds the real paging space available in the system). Thus, the paging manager must keep statistics on system and virtual machine paging activity and make these statistics available to the dispatcher to detect and prevent a potential thrashing condition.

Inefficiency in the second area causes an unacceptable ratio of CP overhead to virtual machine program time, and in extreme cases may cause CP to use excessive processor time. To understand how allocation is determined by CP, the way in which the inventory of real storage page frames is described to the system must be understood.

Each page frame (4096-byte block) of real storage in the system is in one of two basic states: nonpageable or pageable. A nonpageable page must remain resident in real storage for some period of time; thus, the page frame cannot be taken from its current owner to be given to someone else. Pages can be either permanently or temporarily nonpageable, depending upon their use.

Temporary locks usually occur when an I/O operation has been initiated that is moving data either to or from the page, and the page must be kept in real storage until the operation has completed.

A page can also be temporarily nonpageable if it contains an active nonresident CP routine.

In addition, a page can be nonpageable through use of the LOCK command. Pages locked this way are permanently resident until they are explicitly unlocked by the UNLOCK command. Pages that are usually considered permanently nonpageable are those that contain the resident portion of CP and those that contain the system's free storage area in which control blocks, I/O buffers, etc., are built.

The data area that page management routines use to control and allocate real storage is the CORTABLE. Each page frame of real storage has a corresponding entry in the CORTABLE, and because the table entries are fixed in length and contiguous, the entry for any given real page frame may be located directly by indexing into the table. Each entry contains pointers that indicate both the status and ownership of the real page that it represents. Some pointers link page table and swap table entries to the real page (and thus establish ownership), while others link the entry into one of several lists that the paging routines use to indicate the page frame's status and availability for paging. A given CORTABLE entry may appear on either of two lists if its real page frame is available for paging; however, if the page referenced is locked or is in transit, its entry is not in either list and is not referenced when available page frames are being searched for swap candidates. The lists are known as the free list (FREELIST) and the flush list (FLUSHLST), and they represent various levels of page frame availability.

- The free list contains page frames that are immediately available for assignment to a requesting virtual machine. The virtual storage pages for which they were last used have either been released by their owners or they have been paged out to auxiliary storage. Requests for real storage are always satisfied from the free list. If the list has been depleted, the requestor waits until a new page frame becomes available as the result of a virtual storage release or a swap-out.

- The flush list contains page frames that belong to those virtual machines that have been dropped from an active dispatching queue. The flush list is the first place that the page frame selection routine looks to find a page to swap out or to assign to the free list for a virtual machine that requires real storage space.

- The scheduler aids the page selection algorithm by notifying it of virtual machines that are no longer eligible for dispatching (either because they have completed or because they are being held suspended in the eligible list). The scheduler calls the page reset routine when a virtual machine is dropped from a queue and does not immediately reenter the dispatch list. Under heavy paging loads, it is the responsibility of the page reset routine to group all in-storage virtual pages belonging to the virtual machine on an available (or flush) list for easy selection by the page replacement algorithm.

Requests for Real Storage Page Frames

Requests for real storage fall into two general categories; those that are requesting space for a page of virtual storage, and those (such as requests for CP work space) that need page frames for their own use. The former, more general case is discussed first, because the latter case is a subset of the first.

The main page manager routine, DMKPTRAN, maps a request for a specific virtual storage address into a page frame of real storage. This requires that the virtual page be read in and the necessary tables be updated to show the proper status of the page frame.

DMKPTRAN requires that the caller supply only the virtual address to be translated and any options that apply to the page to be located. Most calls are made via the TRANS macro, which sets up the necessary parameters, determines whether or not the required page is resident, and calls DMKPTRAN if it is not.

When DMKPTRAN receives control, it first tests to see if the requested page is resident. This is done via the LRA instruction. If the page is resident, the routine locks the page if requested and exits to the caller. If the LRA indicates that the page is unavailable, it is still possible that the required page is resident. This occurs if the page frame has been placed on the FREELIST but has not been assigned to another virtual machine. When the page swap routine removes a page frame from a virtual machine, the unavailable bit is set in the corresponding page table entry; however, the real main storage index for the page frame is left unchanged. The page table entry is set to zero only when the corresponding page is actually assigned to another virtual machine. Thus, if DMKPTRAN finds the page unavailable, a further test is made on the page table entry to see if the page can be reclaimed. If the entry is not zero (aside from the unavailable bit), the CORTABLE entry for the page frame is removed from the FREELIST and the page frame is returned to the calling virtual machine.

If the page table entry corresponding to the requested virtual page is zero, the required page is not in real storage and must be paged in. However, it is possible that the page is already on its way into main storage. This condition is indicated by a flag in the SWPTABLE entry for the virtual page. The DMKPAGIO routine maintains a queue of CPEXBLOKs to be dispatched when the pending page I/O is complete. The CPEXBLOK for the page in transit is located and a new CPEXBLOK, representing the current request, is chained to it.

Before exiting to wait for the paging operation to complete, DMKPTRAN checks to see if the deferred return (DEFER option) has been specified. If it has not, DMKPTRAN returns to the caller. If the DEFER option has been requested, DMKPTRAN exits to the dispatcher to wait for page I/O completion. When the requested page has been read into real storage, the list of CPEXBLOKs are unstacked first in first out to satisfy all requests for the page that arrived while it was in transit.

If a page is not in transit, a page frame of real storage must be allocated to fill the request. Before the allocation routine is called, a test is made to see if the caller wishes the return to his routine or to be delayed until after the requested page is available. If the DEFER option is not requested, DMKPTRAN returns to the caller after first building and stacking a CPEXBLOK that allows processing of the page request to be continued the next time the dispatcher (DMKDSPCH) is entered.

DMKPTRAN next calls the FREELIST manager (DMKPTRFR) to obtain the address of the next available CORTABLE entry. DMKPTRFR maintains a first-in-first-out list of the CORTABLE entries for those page frames that are immediately available for assignment. As DMKPTRFR releases these page frames, a check is made to see if the number of entries on the FREELIST has fallen below a dynamically maintained minimum value. If it has, the page selection routine (SELECT) is called to find a suitable page frame for placement in the FREELIST. The number maintained as the FREELIST threshold has a value equal to the number of users in queue1 plus the number of users in queue2 plus 1.

The FREELIST is replenished directly by users releasing virtual storage space. The page-out routine, DMKPGSPO, calls DMKPTRFT to place released page frames directly on the FREELIST. However, most replenishment is done via the page selection routine, SELECT. SELECT is called by DMKPTRFR when the FREELIST count falls below the current minimum, or when a user page is reclaimed from the FREELIST. In either case, the selection algorithm attempts to find a page to swap to auxiliary storage. The highest-priority candidates for a swap are those page frames whose CORTABLE entries appear on the FLUSHLST. SELECT attempts to take a flushed page frame before it takes a page frame from an active user. If such a page frame is found, it is checked to see if it has been changed since page-in. If it has not, it is placed in the FREELIST by DMKPTRFT; otherwise, it is scheduled for a swap-out by dequeuing the CORTABLE entry from the FLUSHLST, constructing a CPEXBLOK for dispatching after I/O completion, and exiting to DMKPAGIO by a GOTO. After the paging I/O is complete, the entry is placed on the FREELIST via a call to DMKPTRFT.

If no pages are found on the FLUSHLST, the selection algorithm examines each page in real storage, searching for an available page that does not have its reference bit on. It begins the search at the first available page at the high end of real storage and searches by descending page address. When it reaches the lowest available page address, it starts again from the top of storage. When a page has been found, that page address minus one is checkpointed. The next time the selection algorithm is invoked, it starts from the checkpointed address. As the selection process proceeds, those pages that were not selected have their reference bits turned off. When the selection algorithm is operating in this mode, a virtual page must be referenced at least once per reset cycle (loop around real storage) to avoid selection.

Once a page frame has been selected and page-out is scheduled, control is returned to DMKPTRFR, which then passes control back to DMKPTRAN with the address of the CORTABLE entry that was allocated. In most cases, page-outs are completely overlapped with page-ins. Approximately one half of all page-ins require a corresponding page-out.

Once a page frame has been assigned, DMKPTRAN checks to see if a page-in is required. It usually is, and the DASD address of the virtual storage page must be obtained from the user's swap table entry and the I/O operation scheduled. However, if the page frame has not yet been referenced (as indicated by a DASD address of zero), the real main storage page frame is set to zero, and no page-in is required. After

the page-in operation has been queued, DMKPTRAN exits to the paging I/O scheduler (DMKPAGIO), which initiates the paging operation and exits to the dispatcher (DMKDSPCH) to await the interruption.

Some requests for main storage page frames are handled differently from general virtual-to-real storage mapping. In particular, it may be necessary for CP to obtain additional free storage for control blocks, I/O lists, buffers, etc. This is handled by the free storage manager, which makes a direct call to DMKPTRFR to obtain the needed storage. Usually, this storage is immediately available (due to the page buffering technique previously described). However, if the FREELIST is exhausted, the request for free storage is recognized as a high-priority call and queued first on the list of those waiting for free page frames.

The real storage manager (DMKPTR) accumulates paging statistics that the scheduler (DMKSCH) uses to anticipate user storage requirements. A count of page-reads and page-writes is kept in each virtual machine's VMBLOK; the corresponding total counts for the system are kept in DMKPSA. A running total of the number of pages a virtual machine has resident, at each instance of page-read, is kept in the VMBLOK. A count of the number of times a virtual machine enters page-wait, because a page frame has been stolen from it, is also kept in the VMBLOK. The section entitled "Controlling Multiprogramming" under "Dispatching and Scheduling" describes the use to which the scheduler puts these counts.

VM/370 Virtual=Real Option: The VM/370 virtual=real option involves the mapping in a one-for-one correspondence of a virtual machine storage area with an equivalent real storage area. For instance, virtual page 1 is in real page frame 1 and virtual page 20 is in real page frame 20. Virtual page 0 is relocated at the end of the virtual storage space because it cannot occupy real page frame 0.

The CP nucleus is altered at system generation to support the virtual=real option. Virtual machines with virtual=real (specially identified in the directory) can then log on and use the space reserved for this option. That space can be used by only one virtual machine at a time. Two virtual machines with the virtual=real capability cannot occupy the same space at the same time.

The virtual=real option allows the virtual machine to bypass the control program's CCW translation. This is possible because I/O from a virtual machine occupying a virtual=real space contains a list of CCWs whose data addresses reflect the real storage addresses. The restriction in this situation is that the virtual machine does not perform I/O into page frame 0 because this would perform a data transfer into real page frame 0. At the same time, it is assumed, and cannot be checked, that the virtual machine also does not attempt to do I/O beyond the bounds of its virtual addressing space. To do so would cause the destruction of either the CP nucleus, which resides beyond the virtual machine space, or another user's page.

| If the real I/O device is an MSS 3330V, then CCW translation is not
| bypassed since CP must still be able to recognize an MSS cylinder fault.
| See Appendix B for details.

The bypassing of CCW translation for the virtual machine occupying the virtual=real space is only invoked after the virtual machine has executed the SET NOTRANS ON command. This command can only be issued by the virtual machine occupying the virtual=real space. The command initiates the bypass of CCW translation. This option is automatically turned off if the virtual machine performs an explicit reset or an implied reset by performing a virtual IPL. During virtual machine IPL, I/O must be performed into page frame 0. For this reason, normal virtual IPL simulation assumes CCW translation in effect to accomplish

the full simulation. Once the IPL sequence has completed, CCW translation can be bypassed by issuing the SET NOTRANS ON command.

When the virtual machine demands a page frame through normal use of CP's page tables, the paging routine recognizes the virtual=real capability. It then assigns the virtual page to the equivalent real page frame and does not perform a paging operation, because all these pages are resident and are never swapped out.

Note: The virtual machine running with virtual=real is still run in System/370 relocate mode.

Virtual 270X lines and sense operations from the virtual machine do not use the virtual=real function. These invoke CCW translation for the virtual enable/disable lines and the transfer of the sense bytes.

The UNLOCK command has a VIRT=REAL operand that essentially releases the virtual=real area for normal system paging use. Once the area has been released, it can only be reclaimed for additional virtual=real operations only by an IPL of the VM/370 system. The size of the virtual=real area is an installation specification that is part of the special nucleus generation procedure that is outlined in the VM/370 Planning and System Generation Guide. The size of the area must be large enough to contain the entire addressing space of whatever virtual machine wishes to occupy that space. A virtual machine can use a smaller space than is provided but cannot use a larger space without regenerating the CP nucleus.

DASD STORAGE MANAGEMENT

Any virtual machine's virtual storage pages that have been referenced but are not resident in real storage must be kept in slots on the DASD paging device. DASD page space is assigned only when the page is selected for a page-out. Certain DASD pages may also be marked read-only. Thus, the DASD address slot initially associated with the page should be considered to be the source of the page only. If the page is changed after it has been read into real storage, a new slot must be obtained when it is paged out. Examples of read-only pages are those which contain portions of pageable saved systems and pages which are part of a system spool file. Slots can be reassigned when DMKPTRAN finds that it must swap a page out to a movable-head DASD device. In this case, the old slot is released and the new slot is obtained.

Slot Allocation

If a new slot is required, DMKPGT is called to supply the address of an available slot. DMKPGT maintains a chain of cylinder allocation maps for each cylinder that has been assigned for either virtual storage or spool file paging. The allocation chains for spooling are kept separately from those used for paging so that they can be checkpointed in case of a system failure. However, in other respects they are the same. The allocation blocks for a given volume are chained from the RDEVBLOK for the device on which the volume is mounted. The chains of cylinder and slot allocation blocks are initialized by DMKCPI. Each block on an allocation chain represents one cylinder of space assigned to paging, and contains a bit map indicating which slots have been allocated and which are available. Each block also has a pointer to the next allocation block on the chain, a cylinder number, and a record count. DMKPGT searches this list sequentially until an available slot is found;

its DASD address is then determined and passed back to the calling routine. If DMKPGT cannot find a cylinder with a de-allocated slot, it enters the cylinder allocation phase. When an available cylinder is found, it constructs a page allocation block for this cylinder and allocates a page to the caller.

Cylinder Allocation

DMKPGT controls the paging and spooling I/O load of the system by allocating cylinders evenly across all available channels and devices. In order for a device to be considered available for the allocation of paging and spooling space:

- Its volume serial number must appear in the system's owned list.
- It must have at least one cylinder of temporary space marked as available in the cylinder allocation block which is located on cylinder 0, head 0, record 3.
- It must not be an MSS 3330V volume.

At system initialization time, CPINIT reads in the allocation records for each volume and constructs the chains of device allocation blocks from which DMKPGT allocates the cylinders. In managing the cylinder allocation, DMKPGT takes three factors into consideration: device type, device address, and possible status as a preferred paging device.

A request for a cylinder of virtual storage page space is satisfied by allocating space on a preferred paging device, provided that one exists on the system and that it has page space available. Preferred paging devices are specified by the installation at system generation time, and generally should be devices on which excessive seek times do not occur. A typical preferred paging device would be the IBM 2305 Fixed Head Storage facility. If the 2305 is assigned as a preferred device, it is possible to allocate some of its space for other high-priority data files without excessively degrading paging. An example of such usage would be for high activity read-only saved system pages that are not shared in real storage, and high-activity system residence disks.

It is also possible to designate movable-head DASD devices such as the 3330, 3340, 3350 and 2314/2319 Direct Access Storage facilities as preferred paging devices. The module(s) so designated should not be required to seek outside of a relatively narrow cylinder band around the center of the paging areas. It is advisable to share the access arm of a movable-head preferred paging device with only the lowest-usage data files.

If one or more preferred devices are defined on the system, CP allocates all of the page space available space on these before it allocates on any other available owned volumes. Within the class of preferred devices, space is allocated first on the fastest devices, and these are spread out across channels and devices. Allocation on nonpreferred devices is spread out in the same manner. Cylinders for spooling space are not allocated from preferred devices. Allocation on a given device is done from the relative center of the volume outward, a cylinder at a time in a zig-zag fashion in an attempt to minimize seek times.

When a request to allocate a slot for virtual storage paging is received by DMKPGTGT and the slot must be allocated on a moveable head (2314/2319, 3330, 3340, or 3350) device, a cylinder and slot are selected in the following manner:

1. CP tries to allocate a space on the cylinder at which the arm on the selected device is currently positioned.

2. If slots are not available on the current cylinder, CP tries to allocate space on a cylinder for which paging I/O has been queued.

3. If the above conditions cannot be met, CP allocates space as close to the center of the volume as is possible.

Before DMKIOSQR is called, the queue of IOBLOKs currently scheduled on the device is examined. If paging I/O has already been scheduled on a device, the paging channel programs are slot-sorted and chained together with TICs.

PAGING I/O

DMKPAGIO handles all input/output requests for virtual storage and spooling pages. DMKPAGIO constructs the necessary task blocks and channel programs, expands the compressed slot addresses, and maintains a queue of CPEXBLOKs for pages to be moved. Once the I/O scheduled by DMKPAGIO completes, it unchains the CPEXBLOKs that have been queued and calls DMKSTKCP to stack them for execution. DMKPAGIO is entered by a GOTO from:

- DMKPTRAN to read and write virtual storage pages
- DMKRPA to read and write virtual storage spool buffers

In either case, all that needs to be passed to DMKPAGIO is the address of the CORTABLE entry for the page that is to be moved, the address of a SWPTABLE entry for the slot, a read or write operation code, and the address of a CPEXBLOK that is to be stacked for dispatching after the I/O associated with the page has completed. DMKPAGIO obtains an IOBLOK and builds a channel program to do the necessary I/O, and uses the device code that is part of the page address to index into the system's OWNDLIST and locate the real device to which the I/O request should be directed. If the device is capable of rotational position sensing, the required sector is computed and a SET SECTOR command is inserted into the channel program. The real SIO supervisor DMKIOSQR is then called to schedule the operation on the proper device.

When the interruption for the paging operation is processed by the primary I/O interruption handler, the IOBLOK that controls the operation is unstacked to the interruption return address, waitpage, in DMKPAGIO. waitpage then unchains the CPEXBLOKs that are queued to DMKPAGQ, and then stacks the queued CPEXBLOKs, by calls to DMKSTKCP, in the order in which they were received. The address of the real page frame is filed into the appropriate page table entry and the pointers denoting the ownership of the real page frame are filed into the CORTABLE entry by the processing routines in DMKPTRAN. If a fatal I/O error occurred for the related page frame, the CPEXBLOKs associated with it are flagged, and the dispatcher, DMKSDPCH, sets a nonzero condition code when it activates the pending task. The error recovery followed depends on the operation being performed. Paging I/O errors associated with spooling operations are discussed in "DASD Errors During Spooling" in this

section, while errors associated with virtual storage paging operations are discussed later in the section "Virtual Storage Paging Error Recovery".

DMKPAGIO maintains its own subpool of preformatted paging IOBLOKs. As I/O operations complete, their IOBLOKs are added to a list of available blocks; as new blocks are needed, they are taken from this list. If the list is empty, DMKFREE is called to obtain storage for a new block. DMKPAGIO also periodically calculates system paging overhead. After 200 pages have been moved (read or written), the elapsed time for the 200 page moves is computed, and the paging rate is calculated in page moves per second. The recent paging load, expressed as the percentage of time that more than one half of the system's pages were idle due to page-wait, is averaged with the previous load and re-projected as the expected load for the next interval.

PAGING SUBSYSTEM

The paging subsystem has three major components that have resource optimization algorithms associated with them:

- The page replacement and page selection algorithm that manages the allocation of real storage frames and selects which virtual page to replace.
- An algorithm for the allocation of DASD backing store pages.
- An algorithm for ordering the queue of page I/O reqests.

PAGE REPLACEMENT AND PAGE SELECTION ALGORITHM

VM/370 is a demand paging system. Programs run in virtual storage and when a storage reference is made to a virtual page not currently in real storage, a page fault occurs. A page fault is a program interruption that occurs when a page marked "not in real storage" is referred to by an active page. This page fault represents a demand for a real storage frame in which to place the virtual page. The page replacement algorithm chooses which real storage frame will be allocated to fulfill such a demand. If all real frames in real storage are occupied by other virtual pages, a real frame can only be obtained by replacing one of those virtual pages. The selection of which virtual page to replace is carried out by the page selection algorithm.

The scheduler aids the page selection algorithm by notifying it of virtual machines that are no longer eligible for dispatching (either because they have been dispatched, or because they are being held suspended in the eligible list). The scheduler calls the page reset routine when a virtual machine is dropped from a queue and reset routine when a virtual machine is dropped from a queue and does not immediately reenter the dispatch list. Under heavy paging loads, it is the responsibility of the page reset routine to group all in-storage virtual pages belonging to the virtual machine; it groups them on an available (or flush) list for easy selection by the page replacement algorithm.

The page reset routine cycles through the virtual machine's segment table looking for valid segment entries. When it finds a valid entry, it turns on the segment table entry invalid flag and the page reset routine begins to process the page table associated with that segment table entry. The page table header is timestamped, and if it is a

shared segment, the active segment table entry count is decreased. For a shared segment, if the active count is still greater than zero, no further processing is done. If the count has decreased to zero, for a shared segment, processing continues as if it were a private segment. Each page table entry in a segment is then examined for an in-storage page. If one is found, it has its reference bit reset to zero. In addition, if the heavy paging condition flag has been set, the page table entry is marked invalid, and the real page is placed on the flush list in last-in-first-out order.

Page Selection

The page replacement/page selection algorithm must find a real frame to satisfy a demand for a virtual page. It first attempts to satisfy the demand with a page from the flush list. The flush list contains virtual pages (if any) that belong to virtual machines that are not eligible for dispatching, and therefore are not being used.

Note: A virtual machine may reenter the dispatch list after its pages have been placed on the flush list. If the virtual machine attempts to access any of those pages, they will be reclaimed. The pages are placed on the flush in last-in-first-out order under the assumption that the longer they remain on the list, the higher the probability the virtual machine will reenter the dispatch list and reclaim them.

If no pages are found on the flush list, the selection algorithm examines each virtual page in real storage, searching for an available page that does not have its reference bit on. It begins the search at the first available virtual page at the high end of real storage and searches by descending page address. When it reaches the lowest available page address, it starts again from the top of storage. When a page has been found, that page address minus one is checkpointed. The next time the selection algorithm is invoked, it will start from the checkpointed address. As the selection process proceeds, those pages that were not selected have their reference bits turned off. When the selection algorithm is operating in this mode, a virtual page must be referenced at least once per reset cycle (loop around real storage) to avoid selection.

BACKING STORE ALLOCATION ALGORITHM

There are two parts to the algorithm for allocation of a DASD page record. The first is to find the optimal device on which to allocate a record. The second is then to optimize the record allocation on a particular device.

Device Selection

CP maintains the DASD device chain in two parts. The major part is the ordering of all devices by type and by the TEMP/PAGE classification. All PAGE devices are ordered before all TEMP devices. The device type ordering is: 2305, 3350, 3340, 3330, and 2314. All devices of the same type are chained together off the primary chain. CP attempts to allocate a page record on the highest-level device until all devices at that level are full and then it tries the next lower device type. Within a particular device type, CP allocates records in a round-robin manner, attempting to evenly distribute the allocated records.

Cylinder Selection

Once a device is selected, CP must determine on which cylinder to allocate a record on that device. CP maintains a chain of cylinder record maps, one for each allocatable cylinder on the device. For 2305 devices, CP attempts to keep cylinder map blocks at the head of the chain. The only optimization done for a 2305 is an attempt to minimize the amount of processor time involved in the allocation process. For movable-arm DASD (that is, not 2305), CP attempts to allocate the first available record found when scanning the cylinder map chain.

Page Selection Routine Support

Whenever a changed page is selected for replacement, it must first be copied onto DASD before the real page can be made available. In cases where there is already a DASD record allocated for the page and it is on a movable-arm DASD, the page selection routine deallocates the old record and requests that a new record be allocated. This occurs each time a page is to be written and its current backing-store location is on a movable-arm DASD. Although this represents overhead in terms of processor use, it is justified because it should minimize arm movement and reduce page wait time.

PAGE I/O REQUEST QUEUEING ALGORITHM

The ordering of page I/O requests that are chained together for initiation with one SIO is done on a priority ordering basis. The priority is:

1. In-queue requests
2. Not-in-queue requests
3. Reads
4. Writes
5. Q1 requests
6. Q2 requests

PCI flags are set for page I/O requests. For non-2305 requests, there is an interruption after each request. For 2305 requests, the PCI flag is set so that there is one interruption for each revolution of the drum (one interruption for every three requests).

Note: For installations that are much more constrained by a page I/O bottleneck (as opposed to processor bottleneck), the 2305 PCI mode can be changed to operate in the same way as the non-2305 processing, that is, by allowing an interruption immediately after each request. The SET SRM PCI DISK command causes the PCI flag to be set so there is one interruption for each 2305 page request. SET SRM PCI DRUM changes it back to the default mode of operation.

VIRTUAL STORAGE PAGING ERROR RECOVERY

Errors encountered during virtual storage (as opposed to spooling) paging operations can generally be classified as either soft or hard errors. Soft errors allow the system to continue operation without delay or degradation. Hard errors can cause noticeable effects such as the

abnormal termination of user tasks (abend) and response degradation. Errors that are successfully retried or corrected are known only to the I/O supervisor and the I/O error retry and recording routines; they appear to the second level interruption handlers (such as WAITPAGE) as if the original operation completed normally.

SOFT ERROR RECOVERY: An I/O error that occurs on a page swap-out is considered to be a soft error. DMKPTRAN calls DMKPGTPG to assign a different DASD page slot and the page is re-queued for output. The slot that caused the error is not de-allocated, and thus is not assigned to another virtual machine. All other uncorrectable paging errors are hard because they more drastically affect system performance.

HARD ERROR RECOVERY: Hard paging errors occur on either I/O errors for page reads or upon exhausting the system's spooling and paging space. Recovery attempted on hard errors depends upon the nature of the task for which the read was being done. If the operation was an attempt to place a page of a virtual machine's virtual storage into real storage, the operation of that particular virtual machine is terminated by setting the page frame in error to zero and placing the virtual machine in console function mode. The user and operator are informed of the condition, and the page frame causing the error is not de-allocated, thereby ensuring that it is not allocated to another user.

The control program functions that call DMKPTRAN (such as spooling, pageable control program calls, and system directory management) have the option of requesting that unrecoverable errors be returned to the caller. In this case, the CP task may attempt some recovery to keep the entire system from terminating (abend). In general, every attempt is made to at least allow the operator to bring the system to orderly shutdown if continued operation is impossible.

Proper installation planning should make the occurrence of a space exhaustion error an exception. An unusually heavy user load and a backed-up spooling file could cause this to happen. The operator is warned when 90% of the temporary (paging/spooling) space in the system is exhausted. He should take immediate steps to alleviate the shortage. Possible remedies that exist include preventing more users from logging on and requesting users to stop output spooling operations. More drastic measures might include the purging of low-priority spool files. If the system's paging space is completely exhausted, the operation of virtual machines progressively slows as more and more users have paging requests that cannot be satisfied and operator intervention is required.

VIRTUAL RELOCATION

CP provides the virtual machine the capability of using the DAT feature of the real System/370. Programming simulation and hardware features are combined to allow usage of all of the available features in the real hardware, (that is, 2K or 4K pages, 64K or 1M segments).

For clarification, some term definitions follow:

First-level storage: The physical storage of the real CPU, in which CP resides.

Second-level storage: The virtual storage available to any virtual machine, maintained by CP.

Third-level storage: The virtual storage space defined by the system operating in second-level storage, under control of page and segment tables which reside in second-level storage.

Page and segment tables: Logical mapping between first-level and second-level storage.

Virtual page and segment tables: Logical mapping between second-level and third-level storage.

Shadow page and segment tables: Logical mapping between first-level storage and third-level storage.

A standard, nonrelocating virtual machine in CP is provided with a single control register, control register zero that can be used for:

- Extended masking of external interruptions
- Special interruption traps for SSM
- Enabling of virtual block multiplexing

A virtual machine that is allowed to use the extended control feature of System/370 is provided with a full complement of 16 control registers, allowing virtual monitor calls, PER, extended channel masking, and dynamic address translation.

An extension to the normal virtual-machine VMBLOK is built at the time that an extended control virtual machine logs onto CP. This ECBLOK contains the 16 virtual control registers, 2 shadow control registers, and several words of information for maintenance of the shadow tables, virtual CPU timer, virtual TOD clock comparator, and virtual PER event data. The majority of the processing for virtual address translation is performed by the module DMKVAT, with additional routines in DMKPRG, DMKPRV, DMKDSP, DMKCDB, DMKLOG, DMKUSO, and DMKPTR. The simulation of the relocation-control instructions (that is, LCTL, STCTL, PTLB, RRB, and LRA) is performed by DMKPRV. These instructions, with the exception of LCTL and STCTL, are not available to virtual machines which are not allowed the extended control mode.

When an extended-control virtual machine is first active, it has only the real page and segment tables provided for it by CP and operates entirely in second-level storage. DMKPRV examines each PSW loaded via LPSW to determine when the virtual machine enters or leaves extended control or translate mode, setting the appropriate flag bits in the VMBLOK. Flag bits are also set whenever the virtual machine modifies control registers 0 or 1, the registers that control the dynamic address translation feature. DMKDSP also examines PSWs that are loaded as the result of interruptions to determine any changes in the virtual machine's operating mode. The virtual machine can load or store any of the control registers, enter or leave extended control mode, take interruptions, etc., without invoking the address translation feature.

If the virtual machine, already in extended control mode, turns on the translate bit in the EC mode PSW, then the DMKVATMD routine is called to examine the virtual control registers and build the required shadow tables. (Shadow tables are required because the real DAT hardware is capable of only a first-level storage mapping.) DMKVATMD examines virtual control registers 0 and 1 to determine if they contain valid information for use in constructing the shadow tables. Control register zero specifies the size of the page and segment the virtual machine is using in the virtual page and segment tables. The shadow tables constructed by DMKVATMD are always in the same format as the virtual tables.

The shadow segment table is constructed in first-level storage and initialized to indicate that all segments are unavailable. Flags are maintained in the VMBLOK to indicate that the shadow tables exist. DMKVATMD also constructs the shadow control registers 0 and 1. Shadow control register 0 contains the external interruption mask bits used by CP, mixed with the hardware controls and enabling bits from virtual control register 0. Shadow control register 1 contains the segment table origin address of the shadow segment table.

When the virtual machine is operating in virtual translate mode, CP loads the shadow control registers into the real control registers and dispatches the user. The immediate result of attempting to execute an instruction is a segment exception, intercepted by DMKPRG and passed to DMKVATSX. DMKVATSX examines the virtual segment table in second-level storage. If the virtual segment is not available, the segment exception interruption is reflected to the virtual machine. If the virtual segment is marked available, then DMKVATSX:

- Allocates one full segment of shadow page table, in the format specified by virtual control register 0.
- Sets all of the page table entries to indicate page not in storage.
- Marks the segment available in the shadow segment table.
- Redispatches the virtual machine via DMKDSP.

Once again, the immediate result is an interruption, which is a paging exception and control is passed to DMKVATPX. DMKVATPX references the virtual page table in second-level storage to determine if the virtual page is available. If the virtual page is not available, the paging interruption is reflected to the virtual machine. However, if the virtual page is marked in storage, the virtual page table entry determines which page of second-level storage is being referenced by the third-level storage address provided. DMKVATPX next determines if that page of second-level storage is resident in first-level storage at that time. If so, the appropriate entry in the shadow page table is filled in and marked in storage. If not, the required page is brought into first-level storage via DMKPTRAN and the shadow page table filled in as above.

As the virtual machine continues execution, more shadow tables are filled in or allocated as the third-level storage locations are referenced. Whenever a new segment is referenced, another segment of shadow page tables is allocated. Whenever a new page is referenced, the appropriate shadow page table entry is validated, etc. No changes are made in the shadow tables if the virtual machine leaves translate mode (usually via an interruption), unless it also leaves extended control mode. Dropping out of EC mode is the signal for CP to release all of the shadow page and segment tables and the copy of the virtual segment table.

There are some situations that require invalidating all of the shadow tables constructed by CP or even releasing and reallocating them. Whenever DMKPTR swaps out a page that belongs to a virtual relocating machine, it sets a bit in the VMBLOK indicating that all of the shadow page tables must be invalidated. Invalidation of all of the tables is required since CP does not know which third-level storage pages map into the second-level page that is being swapped out. The actual invalidation is handled by DMKVATAB, called from DMKDSP when the virtual machine is on the verge of being dispatched.

The other situations which cause shadow table invalidation arise from the simulation of privileged instructions in DMKPRV. Flags are set in the VMBLOK whenever the virtual machine loads either control register 0 or 1, and DMKPRV calls DMKVATAB to perform whatever maintenance is required. When control register 1 is loaded by the virtual machine, DMKVATAB must re-copy the virtual segment table into first-level storage and invalidate the entire shadow segment table. When control register 0 is loaded, DMKVATAB examines the relocation-architecture control bits to determine if they have changed, (such that the format of the virtual page and segment tables no longer matches that of the shadow tables). If the format has not changed, the shadow tables are left intact; otherwise, all of the shadow tables must be returned to free storage and another set, in the new format, must be allocated and initialized. The same actions can result from modifying the control registers via the CP console functions, in which case DMKVATAB is called from DMKCDB. The privileged operation, PTLB, also causes the virtual segment tables to be re-copied and all of the shadow page tables to be invalidated because the shadow tables are the logical equivalent of the translation look-aside buffer.

DMKPRV provides virtual interrogation of the reference and change bits in the virtual storage keys, which involve the privileged instructions ISK, SSK, and RRB. The privileged instruction LRA is simulated via DMKVATLA, which searches the virtual page and segment tables to translate a third-level storage address to a second-level storage address, returning a condition code indicator to DMKPRV, or forcing an interruption if the tables are incorrectly formatted.

Most error situations that occur in the virtual machine are handled by means of the extended program interruptions associated with the real address translation hardware. Whenever a virtual relocating machine loads control registers 0 or 1 with an invalid value, DMKVAT releases all of the shadow tables exactly as if the hardware controls had changed. The shadow control registers are set valid, with the shadow segment table re-allocated at a minimum size and all segments marked unavailable. Flag bits are set in the VMBLOK to indicate that the shadow tables are artificially valid, and DMKVATSX reflects a translation specification exception to the virtual machine as soon as it is dispatched. While it is possible for the virtual machine to enter an interruption loop (if the new PSW is also a translate mode PSW), the cited process prevents the occurrence of a disabled loop within CP, which would result if the virtual machine is never dispatched.

FREE STORAGE MANAGEMENT

DMKFRE is responsible for the management of free storage, and CP uses it to obtain free storage for I/O tasks, CCW strings, various I/O buffers, etc. It is used, in fact, for practically all such applications except real channel, control unit, and device blocks, and the CORTABLE.

Block sizes of 30 doublewords or less, constituting about 99 per cent of all calls for free storage, are grouped into 10 subpool sizes (3 doublewords each), and are handled by LIFO (push-down stack) logic. Blocks of greater than 30 doublewords are strung off a chained list in the classic manner.

When subpools are exhausted, small blocks are generally obtained from the first larger block at the end of available free storage. Large blocks, on the other hand, are obtained from the high-numbered end of the last larger block. This procedure tends to keep the volatile small subpool blocks separated from the large blocks, some of which stay in storage for much longer periods of time; thus, undue fragmenting of available storage is avoided.

DMKFRE initially starts without any subpool blocks. They are obtained from DMKFREE and returned to DMKFRET on a demand basis.

The various cases of calls to DMKFREE for obtaining free storage, or to DMKFRET for returning it, for subpool sizes and large sizes, are handled as follows:

Calling DMKFREE for a Subpool

Subpool Available: If a call for a subpool is made and a block of the suitable size is available, the block found is detached from the chain, the chain patched to the next subpool block of the same size (if any), and the given block returned to the caller.

Subpool Not Available: If a block of suitable size is not available when a call to DMKFREE is made for a subpool, the chained list of free storage is searched for a block of equal or larger size. The first block of larger or equal storage is used to satisfy the call (an equal-size block taking priority), except that blocks within the dynamic paging area are avoided if at all possible. If no equal or larger block is found, all the subpool blocks currently not in use are returned to the main free storage chain, and then the free storage chain is again searched for a block large enough to satisfy the call. If there still is no block large enough to satisfy the request, then DMKPTRFR is called to obtain another page frame of storage from the dynamic paging area, and the process is repeated to obtain the needed block.

Calling DMKFREE for a Large Block

If a call to DMKFREE is made for a block larger than 30 doublewords, the chained list of free storage is searched for a block of equal or larger size. If an equal-size block is found, it is detached from the chain and given to the caller. If at least one larger block is found, the desired block size is split off the high-numbered end of the last larger block found, and given to the caller. If no equal or larger block is found, DMKPTRFR is called to obtain another page frame of storage from the dynamic paging area, and the above process is repeated (as necessary) to obtain the needed block.

Calling DMKFRET for a Subpool

If a subpool block is given back via a call to DMKFRET, the block is attached to the appropriate subpool chain on a LIFO (push-down stack) basis, and return is made to the caller. If, however, the block was in a page within the dynamic paging area, the block is returned to the regular free storage chain instead.

Calling DMKFRET for a Large Block

If a block larger than 30 doublewords is returned via DMKFRET, it is merged appropriately into the regular free storage chain. Then, unless the block was returned by DMKFRETR (see "Initialization") a check is made to see if the area given back (after all merging has been done) is a page frame within the dynamic paging area. If so, DMKPTRFT returns it to the dynamic paging area for subsequent use.

Free Storage Page Frame Allocation

The number of page frames allocated to free storage depends upon:

1. The real machine storage size
2. The RMSIZE operand specified in the SYSCOR macro at system generation time
3 The FREE operand in the SYSCOR macro

The storage size used by VM/370 is the smaller of the real machine storage size and the RMSIZE value.

If the FREE operand was not included in the SYSCOR macro statement for DMKSYS, the default number of fixed free storage pages allocated at IPL time for the first 256K of storage is 3 and 1 page for each 64K thereafter, not including V=R size, if any.

If the FREE operand was included in the SYSCOR macro statement for DMKSYS, that value is the number of fixed free storage page frames allocated at IPL time. If those pages represent an amount of free storage greater than 25% of the VM/370 storage size (not including V=R size, if any) the default allocation is used.

CP INITIALIZATION

System initialization starts when the operator selects the DASD device address of the CP system residence volume (SYSRES) and presses the IPL button. The System/370 hardware reads 24 bytes from record 1 of cylinder 0 on SYSRES into location 0 of main storage. This record consists of an initial PSW and a channel program. The channel program reads the module DMKCKP into location X'800' and gives it control. DMKCKP checks location CPID in module DMKPSA.

If CPID contains the value CPCP or WARM, DMKCKP saves the spool file control blocks, system log messages, accounting information, status of spool devices, spool hold queue blocks, and spool record allocation blocks and writes them on the warm start cylinders. If CPID contains the value CPCP, DMKCKP loads a disabled wait state code X'008'.

If location CPID does not contain the value CPCP, DMKCKP now loads DMKSAV and passes control to it at entry point DMKSAVRS. DMKSAV reloads a page image copy of the CP nucleus into real storage starting at page 0. When DMKSAV is finished, control is transferred to DMKCPI. DMKCPI performs the main initialization function. This includes calling DMKWRM to retrieve the information stored on the warm start cylinder. This also includes calling DMKCKS to initialize the dynamic checkpoint cylinders and to checkpoint the current status of the spool file system. When DMKCPI has finished, it passes control to DMKDSPCH. DMKDSPCH loads a wait state PSW to wait for work. See "CP Initialization for the Attached Processor" for additional information.

Attaching a Virtual Machine to the System

After CP has been initialized, DMKCPVEN enables the communication lines in response to the ENABLE command. Then an individual virtual machine is attached to the system, using the following steps:

1. Terminal Identification

   When the CP receives the initial interrupt from a terminal on an enabled line (normally initiated by a user dialing in on a data-set), the DMKCNSIN routine is entered. DMKCNSIN determines the terminal device type, stores this information in the terminal device block, writes the online message and puts the terminal line in a state to receive an attention interruption.

2. Attention from User

   After the online message has been displayed at the user's terminal, and he has pressed the ATTENTION key, DMKCNSIN (the console interruption routine) calls DMKBLDVM to build a skeleton VMBLOK for the user. At this time, the userid is LOGONxxx, where xxx is the terminal real device address, and a flag is set to indicate that the user has not yet completed the logon process.

   Then DMKCNSIN calls DMKCFMBK, which types a single blank at the terminal, and issues a read to the terminal for the user to enter his first command (normally LOGON or DIAL).

3. First Command from User

   After the first command has been entered by the user, DMKCNSIN further determines the type of terminal. If the terminal is a 2741, DMKTRMID is called to identify it as either a 2741P (PTTC/EBCD) or a 2741C (Correspondence) terminal. If successful, the correct device type and translate tables for input and output are set; if not, flags are set to indicate that the terminal is not yet identified.

   Then control is returned to DMKCFMBK, which determines if the first command is valid (for example, LOGON, MSG, or DIAL). If the first command is not valid, a restart message is given, and the read to the terminal occurs again for the first command. If the first command was LOGON (or its abbreviation), DMKLOGON is called to complete the process of attaching the virtual machine to the system.

   The operations performed by DMKLOGON include the following:

   - Ensures that the maximum number of virtual machines allowed on the system is not being exceeded.
   - Obtains the userid from the command line, and checks for a possible password and other optional operands.
   - Checks the userid and password (entered separately if not on the LOGON command line) against entries in CP's directory of users.

- Ensures that the user is not logged on at another terminal (an error condition), or reconnects the user if he was running in disconnect mode.

- Obtains pertinent information on the user's virtual machine from the user machine block portion of the directory.

- Stores the correct userid (replacing the LOGONxxx userid used until now), virtual storage size, and other vital information in the virtual machine's VMBLOK.

- Allocates and initializes segment, page, and swap tables (necessary for handling of the virtual machine's virtual storage).

- Schedules MSS volume mounts for any required MSS volumes if the MSS is available and the volume is not already mounted.

- Allocates an extended VMBLOK (ECBLOK) if the user's virtual machine has the ability to run in the extended control mode.

- Allocates and initializes virtual device blocks, control unit blocks, and channel blocks, using information from the user device blocks portion of the directory.

- Establishes links (as feasible) to all DASD devices included in the directory, the accessibility of any disk being determined by the user access mode in the directory, and whether any other users are presently linked to the disk, in read mode and/or write mode.

- Initializes all other virtual device blocks as appropriate, such as reader, punch, printer, and terminal.

- Maps all virtual devices to real devices.

- Performs appropriate accounting.

- Informs the user of the date and time of the most recent revision to the system log message (LOGMSG), and of the presence of any outstanding spooled files in his virtual reader, printer, or punch.

- Sends a ready message to the user with the date and time (and weekday), and a message to the system operator indicating that the user has logged on.

If the virtual machine has a device address or a named system in the directory and the initialization was not suppressed via an option on the LOGON command line, then that device or named system is then loaded (via IPL) at the conclusion of the logon process. Otherwise, when the logon functions are complete, the user's terminal is placed in CP read mode ready for the entry of his first desired command.

Under the latter condition of no automatic IPL, the user can IPL an alternate nucleus by using the STOP option in the IPL command. This option causes the normal IPL procedure to halt execution prior to loading the initial PSW, and issues a DIAGNOSE code 8 that places the user's terminal in CP read mode. A hexadecimal character entered in location X'08' changes the nucleus name. A hexadecimal character entered in location X'09' changes the apparent storage size. The BEGIN command allows the IPL procedure to continue.

I/O Reconfiguration

Three commands alter the I/O configuration of a user's virtual machine after he has logged on. Two are user commands, while the third a system operator command, because it affects the status of real devices attached to the system. The ATTACH and DETACH commands are contained in DMKVDA, DMKVDC, DMKVDD, and DMKVDE and the DEFINE command in DMKDEF. The system command scanner (DMKCFM) calls both pageable modules after their format and privilege classes have been validated. These commands access the same control-block building subroutines in the module DMKVDS that DMKLOG, the LOGON processor, uses.

Attaching a Real Device: The system operator can dedicate any real device to a single virtual machine by issuing the ATTACH command. The device attached is available only to the given virtual machine, and all I/O requests to it are handled by CCW translation. If the device is a DASD, cylinder relocation does not occur when SEEK addresses or home addresses are referenced. The I/O supervisor does not queue operations on the device, nor does it automatically restart it or do ordered seek queuing. Nonsharable devices such as tape drives must be attached to a virtual machine to be accessed by the virtual machine. A virtual machine can also have a dedicated card reader/punch or printer. However, this is usually not necessary because of the unit record spooling facilities of CP. Unit record input or output on a dedicated (attached) device is not spooled by CP. The unit attached may be given a virtual address different from its real address; however, the virtual machine may not already have a virtual device at the attached address. A real device cannot be attached (1) if it is currently dedicated to another virtual machine, (2) if it contains minidisks that are in use by other vitual machines, or (3) if it is a system-owned volume that is in use for spooling or paging.

Defining a Virtual Device: A system user can define a new virtual device with the DEFINE command that does not require the dedication of a corresponding real device. Devices that can be defined are consoles, spooled readers, punches and printers, dialable TP lines, virtual channel-to-channel adapters, pseudo timers, and temporary disks. With the DEFINE command, the user can change any existing virtual device address whether it corresponds to a shared or dedicated real device or no real device unit.

The DEFINE command can also describe the virtual machine channel mode of operation, that is, either selector or block multiplexer. The default mode, selector channel mode, reflects a channel busy to any SIO operation attempted on the same channel path that has not completed the previous channel SIO operation. Block multiplexer mode allows the successful initiation of different devices on the same channel path. Channel 0, a byte-multiplexer channel, is unaffected by the DEFINE command. Also, any channel with a channel-to-channel adapter (CTCA) defaults to selector mode of operation regardless of the channel mode selected. Use of the DEFINE command with the CHANNELS operand generates a virtual machine reset; therefore, it should be invoked prior to the virtual machine IPL operation.

Note: The channel mode selected has no bearing on the types of channels that are attached to the real system.

Temporary disks are dynamically obtained cylinders of DASD storage space. They are available to the user for as long as they are part of his virtual machine configuration, but the data on them is destroyed after the user detaches the area. For all other purposes, however, they appear to be a standard disk.

Detaching a Virtual Device: A virtual device can be removed from a virtual machine configuration prior to logging off with the DETACH command. A user can detach any of his own devices, and the system operator can detach a real device from a virtual machine. If the operator detaches the device, the user is informed of the operator's action. A real device can be detached only if it is dedicated to a single virtual machine or is attached to the system and is not in use when the DETACH is issued.

Disconnecting a Terminal or Virtual Machine

A user may permanently or temporarily disconnect his terminal or virtual machine from the system by a console command, or the terminal or virtual machine may be forcibly disconnected by the operator. The system can also log off the virtual machine. In any case, the routines that handle the termination process are in the pageable module, DMKUSO.

PERMANENT DISCONNECT: The user may voluntarily remove his virtual machine from the system via the LOGOFF command. This command terminates all virtual machine operation, releases all storage occupied by control blocks and virtual storage pages, and disconnects the teleprocessing line connection to the user's terminal. If the user specifies the HOLD option with LOGOFF, all of the above occurs, except that the teleprocessing line remains enabled. This option is especially useful for dialed connections that are reused immediately by another user.

The virtual machine can be forced off the system by the system operator via the FORCE command. This has the same effect as a user-initiated logoff, except that the user is informed that the operator has logged off his machine. A virtual machine may also be logged off the system:

- If the time for a read of a system password expires (28 seconds).
- If the user makes a connection to the system but does not logon within a given period.
- If the virtual machine is running disconnected (without an active terminal) and the virtual machine attempts a terminal read or enters a disabled wait state.

The DMKUSOLG and DMKUSOFF subroutines process the LOGOFF command. DMKDSP calls DMKUSOFF directly by DMKDSP to force the logoff of a disconnected user as previously described.

TEMPORARY DISCONNECT: A user may temporarily disconnect his terminal from his virtual machine by using the DISCONN command, while allowing the virtual machine to continue to run. This command flags the virtual machine as being disconnected and releases the user's terminal and teleprocessing line. If the HOLD option was specified in the DISCONN command, CP allows the line to remain enabled, and another user can use the terminal to log on. The disconnected virtual machine continues to be dispatched until it either attempts to execute a terminal read to the disconnected console or it enters a disabled wait state. At this time, the dispatcher (DMKDSP) calls the routine DMKUSOFF directly to force the machine out of the system. While the machine is disconnected from its virtual console (real terminal) any terminal output is lost; in addition, CP may apply a disconnected penalty to the machines scheduling priority, to bias the system in favor of interactive users.

A user's virtual machine may also be disconnected by the system. If the disconnected user logs on to the system while the disconnected machine is still running, it is reconnected and can continue to interact with the system in the usual manner.

The DMKUSO subroutine processes the DISCONN command.

CONSOLE FUNCTIONS

DMKCFM analyzes CP commands and passes control to the appropriate routine to handle the command. DMKCFM can be entered by the Attention key (or equivalent) at the user's terminal or directly from a virtual machine.

When a console interruption occurs by the Attention key at the user's terminal, DMKIOSIN calls DMKCNSIN to handle the unsolicited interruption, then DMKCNSIN calls DMKCFMBK.

DMKCFMBK first calls DMKFREE to obtain storage for an 18-doubleword input buffer. Next, DMKQCNWT is called to send the CP message to the terminal to inform the user that he has entered console function mode. DMKQCNRD is then called to read the command line entered at the console.

DMKCFMEN is the entry point for commands coming directly from the virtual machine. DMKPRGIN enters at DMKCFMEN here when a DIAGNOSE instruction with a code of 8 is detected. The address of an 18-doubleword input buffer is passed in register 1; therefore, a read to the terminal is not needed.

After either the read to the terminal or entry from the virtual machine, DMKSCNFD is called to find the command type. On return from DMKSCNFD, register 1 points to the start of the command and register 0 contains the length of the command. DMKCFCMD is then called and the command is matched against a list of valid commands. The list contains a 16-byte entry for each command. Each entry contains 8 bytes for the name, 2 bytes for class mask, 2 bytes for an abbreviation count, and 4 bytes containing the routine address. If the entered command matches an entry in the list, it is then checked to ensure that a valid abbreviation for the command has been used. If this test is not successful, DMKSCN continues to scan the list for a valid command. Should the abbreviation be valid, a check is then made to determine if this user is of the proper class to use the command entered. If this is successful, DMKCFM then calls the appropriate routine to process the command.

After the command has been processed, control is returned to DMKCFM. There are three possible returns. (1) On a normal return, the input buffer is scanned to see if there are any more commands. If none exist, DMKCFM returns to the virtual machine (if entered via DIAGNOSE) or calls DMKQCNRD to read the next command from the terminal. (2) On a return plus 4, the VMCFWAIT bit is turned off to allow the virtual machine to run. DMKFRET is called to return the input buffer storage. Then control returns to either the virtual machine, if entered via a DIAGNOSE, or to DMKDSPCH if entered via the Attention key. (3) On a return plus 8, the operation is the same as plus 4 except that the VMCFWAIT bit is left on.

The scheduler, DMKSCH, selects dispatchable virtual machines from the virtual machine population. The auxiliary routine that assists the scheduler and dispatcher is the request stack maintenance routine, DMKSTK.

To make decisions on dispatching and scheduling, the control program places all virtual machines into various categories, and recognizes user machines as being in one of several states. The virtual machine categories either interactive or noninteractive virtual machine, are defined in the following way:

- An interactive virtual machine is one whose use of the system is punctuated by regular and frequent terminal I/O, and does not have long processor execution times. A virtual machine becomes eligible to enter interactive status whenever a channel program for virtual console I/O has completed, or whenever I/O for a dedicated or dialed virtual telecommunications line has completed.

- A non-interactive virtual machine is one that has violated an interactive criterion, or one that has entered an idle wait state by entering console function mode (equivalent to stopped state), or by loading a wait state PSW that is not enabled for any busy channel. CP schedules interactive users ahead of non-interactive users. Non-interactive users are subdivided into several classes. Normal non-interactive virtual machines are scheduled by a priority scheme described below. A virtual machine is allowed to execute for a specified time period and then it is placed in a list of those machines that are waiting.

To give preference to certain classes of virtual machines, a priority scheduling scheme allows virtual machines to be scheduled with a priority class. The priority is a number assigned by the directory; however, the number may be altered by the system operator.

Virtual Machine Dispatching Lists and States

To efficiently manage the large inventory of potential virtual machines that are logged on to the system, CP defines several states that a virtual machine may occupy. The scheduler can move a virtual machine from one state to another; however, a virtual machine may exist in only one state at any given instant. CP can then make scheduling and dispatching decisions by looking only at the subset of virtual machines that are in the appropriate state. To do this search, it also maintains lists of virtual machines in certain executable states.

A user's virtual machine may be in one of the following states:

State	Meaning
1	Interactive and dispatchable (in queue1, in dispatch list)
2	Interactive and not dispatchable (in queue1, in dispatch list)
3	Interactive and eligible for queue1, but no available real storage (waiting for queue1, in eligible list)
4	In wait state with terminal read or write active
5	Non-interactive and dispatchable (in queue2, in dispatch list)
6	Non-interactive and not dispatchable (in queue2, in dispatch list)
7	Non-interactive and eligible for queue2, but no available real storage (waiting for queue2, in eligible list)
8	Idle - waiting for asynchronous I/O or external interruption, or stopped (in console function mode)

Entries on the dispatch list are the VMBLOKs for those virtual machines in states 1, 2, 5, and 6, and represent the virtual machines that can be run at any given time. (States 2 and 6 remain in the dispatch list even though they are not dispatchable.) The dispatch list is sorted by dispatching priority, which is the ratio of processor time to wait time over the length of the current virtual machine task. A task is defined as that execution that takes place between terminal reads or entry to enabled wait (that is, movement from state 4 or 8 to state 1) and is re-projected for a virtual machine each time it is dropped from a queue. Virtual machines entering state 1 always have a priority of 0.

The eligible list contains virtual machines in states 3 and 7: these virtual machines are potentially executable, but due to the current load on the system they are not allowed to compete for the processor. As soon as a virtual machine in the dispatch list is dropped from queue, the highest-priority virtual machine(s) in the eligible list is added to the dispatch list. Conditions can arise where the virtual machine that is added to the dispatch list has a projected working set size that far exceeds the remaining system capacity. The eligible list has two components; a section composed of those virtual machines waiting for Q1 (interactive) and a section composed of those virtual machines waiting for Q2 (non-interactive). Each section of the list is sorted by scheduling priority, which is determined at the time the virtual machine is added to the eligible list, as follows:

1. The virtual machine's projected working set size, calculated the last time it was dropped from a queue, is expressed as a percentage of the amount of main storage available for paging. This percentage, usually between 0 and 100, is multiplied by the paging bias factor (stored at DMKSCHPB).

2. The virtual machine's priority (the priority set by the directory or the class A SET PRIORITY command) is multiplied by the user bias factor (stored at DMKSCHUB), and is added to the paging bias calculated in step 1.

3. The sum of paging and user bias is divided by the sum of the bias factors to obtain a weighted average.

4. A base priority is obtained by storing the TOD clock and using the leftmost word, which increases by 1 approximately once per second. This word is then modified by shifting it left or right based on the priority delay factor (stored at DMKSCHPD). If DMKSCHPD is positive, it indicates a right shift, thereby increasing the delay interval of the base priority. A negative value indicates a left shift.

5. The weighted average obtained in step 3 is then logically added to the adjusted base obtained in step 4.

6. If the virtual machine is entering Q2 for the first time after being dropped from Q1, the interactive bias factor (stored at DMKSCHIB) is subtracted from the priority obtained in step 5. If the virtual machine is entering Q1, or if it was last dropped from Q2, the interactive bias is not applied.

7. The result of steps 1 through 6 is the scheduling or eligible list priority, and is stored in the VMEPRIOR field of the VMBLOK.

The VMBLOK is then sorted into the appropriate section of the eligible list in ascending value of VMEPRIOR. The effects of the various biases and the delay factor are illustrated by the following examples.

Example 1
Assume that two virtual machines are to be added to the eligible list for Q2. The paging bias factor is 1, the user bias factor is 1, and the priority delay factor is 0. Virtual machine A has a projected working set size of 80 percent of available storage and a user priority of 50. Virtual machine B has a projected working set size of 20 percent of available storage and also has a user priority of 50. The biases are obtained as follows:

User	Paging Bias		User Bias		Weighted Bias
A	80 X 1	+	50 X 1	= 130/2 =	65
B	20 X 1	+	50 X 1	= 70/2 =	35

If A is added to the eligible list at base time 0, its eligible list priority is 65. If the priority delay factor is 0, B is added ahead of A provided that B is eligible for entry to the list within the next (65-35) 30 seconds. If the priority delay factor is set to +1, the base is incremented once every two seconds. Therefore, although the bias difference is still 30, the delay time is now 60 seconds.

Example 2
To force A to be given a weighted bias equal to B, a priority differential is calculated as follows:

$$\frac{80 + A}{2} = \frac{20 + B}{2}$$

$$A = B - 60$$

Therefore, for the biases to be equal, A must have a priority of 60 less than B. For example, if A is given a priority of 10 and B is given a priority of 70, the biases would compute as follows:

User	Paging Bias		User Bias		Weighted Bias
A	80 X 1	+	10 X 1	= 90/2 =	45
B	20 X 1	+	70 X 1	= 90/2 =	45

Example 3
The large difference in priorities could be lessened by increasing the user bias factor. If the user bias factor is set to 3 instead of 1, the calculated priority differential is as follows:

$$\frac{80 + 3A}{4} = \frac{20 + 3B}{4}$$

$$3(B - A) = 60$$

$$A = B - 20$$

Now, A requires a priority of only 20 less than B to achieve parity. For example:

User	Paging Bias		User Bias			Weighted Bias
A	80 X 1	+	30 X 3	=	170/4	= 42
B	20 X 1	+	50 X 3	=	170/4	= 42

The above examples illustrate the following general points about the use of the bias factors, the delay factor, and the user priority value:

1. The paging and user bias factors are a measure of the relative importance of the bias value. A high user bias allows greater discrimination via the assigned priority; while a high paging bias makes storage requirement the primary scheduling parameter.

2. The virtual machine priority value, in the directory, may be overridden, and is the means through which selected users obtain improved performance.

3. The priority delay factor is the measure of the impact that the paging and user biases are to have. The greater the delay value, the greater is the maximum delay that can be experienced by a given user.

4. The interactive bias factor is a tool that enhances command response to conversational commands that require disk I/O, and that may be partially executed in Q2.

If the paging bias factor is nonzero, the net effect of the priority scheme is to discriminate against virtual machines that require large amounts of real storage. This discrimination results in a higher level of multiprogramming and increased processor utilization; however, it must be traded off against poorer throughput for large storage users. The distributed scheduler is not biased; the bias factors are as follows:

```
Paging bias factor      (DMKSCHPB) = 0
User bias factor        (DMKSCHUB) = 1
Priority delay factor   (DMKSCHPD) = 0
Interactive bias factor (DMKSCHIB) = 0
```

Thus, the basic VM/370 scheduler schedules virtual machines FIFO within user priority.

Figure 19 is a graphic breakdown of the user states, showing the relationship between interactive and non-interactive states, in-queue and not-in-queue states, and in-list and not-in-list states.

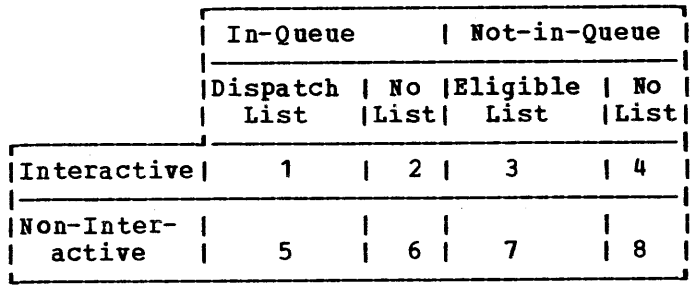


Figure 19. User Dispatching States

Figure 20 shows the possible user-state changes and the reasons for them; any changes not described are not possible.

Status Change		
From	To	Reason for Status Change
1	2	Pagewait, SIO-WAIT, or enabled wait for any busy channel
1	4	Enabled wait for interactive terminal read or write
1	5	Exceeds in-queue time slice
1	7	Same as 1 to 5 except that queue2 is full
1	8	Wait without active I/O, disabled WAIT or hit ATTN
2	1	Wait condition complete
2	5,7	Wait completes, but in-queue time slice exceeded
3	1	Another user drops from queue1 and now there is room
4	1	Terminal I/O completes while user is waiting
4	3	Terminal I/O completes, but queue1 is full
5	1	Terminal I/O completes while user is active in queue2
5	4	User puts up terminal read or write and enters wait
5	6	Pagewait, SIO-WAIT, or enabled wait for busy channel
5	7	Dropped from queue2 due to in-queue time-slice end
5	8	Wait without active I/O, disabled WAIT, or hit ATTN
6	5	Wait condition completes
7	5	Room is found in queue2
8	5,7	Asynchronous I/O or external interruption or BEGIN

Figure 20. User Status Changes

Controlling of Multiprogramming

To control the number of virtual machines allowed in queue, the scheduler monitors the paging activity of all virtual machines and of the total system. A decision as to whether or not to move a potential virtual machine from the eligible to the dispatch list is based upon whether or not that its projected working set exceeds the system's remaining capacity. Individual virtual machines' working sets are calculated and projected at queue drop time according to one of the following formulas:

$$P = (A+P)/2$$

If $(LP-LA) * (P-A) < 0$

-- or --

$P = A$

If $(LP-LA) * (P-A) \geq 0$

where:

A Actual working set at queue drop time

LA Last actual working set

LP Last projected working set

P Current projected working set

The working set is added to the current system load, which consists of the sum of the working sets for all virtual machines currently in a queue. The sum is compared to the system maximum, which is equal to the number of dynamically assignable pages in the system. If the virtual machine's projected working set will not push the system load over the virtual machine maximum, it is placed in the queue and added to the dispatchable list.

The actual working set, A, is the smaller of the two values determined at queue drop time by the following formula:

$$A = \left[\sum_{i=1}^{N} PRi \quad / \; N + Steals \right]$$

-- or --

Pages referenced

where:

N	Number of page reads while in queue.
PR	Number of pages resident at the ith page read.
Steals	Number of times page wait was entered because of a stolen page.

The number of referenced pages is determined by scanning the virtual machine's page tables for software referenced bits. These bits are set by DMKPTRAN when the page is taken from the virtual machine by CP. Thus the actual working set is generally the average number of pages resident at each page read. However, this estimate is sensitive to the overall system paging activity for the following reasons:

1. If there is no paging load on the system, there is one page read for each resident page, and no steals; the working set therefore tends to be equal to about one half of the resident page total.
2. As paging activity increases, and the working set location shifts, the working set tends to increase toward the average number of resident pages.
3. If paging activity becomes excessive, the number of page steals increases to the extent that the working set expands to the maximum of the total number of pages referenced while in the queue.

In summary, the scheduler selects the subset of logged-on virtual machines that are allowed to compete for the resources of the processor, with the constraint that a new virtual machine is not added to the active subset if its projected main storage requirement, added to those of the other active virtual machines, causes the current capacity of the

system to be exceeded. Selection within scheduling priority simply means that a executable virtual machine of high priority is always added to the active subset (to a queue) before a executable virtual machine of lower priority. If the paging bias mechanism is activated by setting the paging bias factor to a nonzero value, scheduler selection is in favor of smaller virtual machines; otherwise, selection is within priority. Once the active subset (the set of in-queue virtual machines) has been selected, the dispatcher allocates resources of the processor among them.

The list of executable virtual machines in a queue is sorted by dispatching (as opposed to scheduling) priority. The dispatching priority is a running average of a given virtual machine's processor time/wait-time ratio. Thus, virtual machines who are most likely to go into wait state, based on past performance, are dispatched ahead of those whose demands on the processor are more extensive. This simple ratio priority is normally altered if a virtual machine is identified as compute-bound by means of the fact that it has executed for at least 50 ms. without entering the wait state. In this case, it is placed at the bottom of the dispatchable list. On the other hand, virtual machines identified as interactive by virtue of the frequency their requests for terminal I/O are placed at the top of the dispatchable list.

Fast Redispatch

DMKDSP also provides a fast dispatch path for virtual machines that have issued specific privileged instructions that are not handled by the Virtual Machine Assist feature.

These virtual machines can be dispatched very rapidly because the virtual machine's program old PSW needs very little reconstruction to redispatch the virtual machine, hence use of full PSW reconstruction path is not required. The decision for using the fast dispatch path (DMKDSPA) is accomplished by the module that handles privileged operation, DMKPRV or DMKVIO. A fast redispatch path is also available after I/O interrupts. If DMKDSP can determine that the I/O interrupt processing had no effect on the running virtual machine's status and it caused no higher-priority virtual machine to become runnable, then the virtual PSW stored at the I/O old PSW location will be used to redispatch the virtual machine.

Enable Window

The CP supervisor runs disabled for all I/O and external interrupts. The dispatcher, in order to alleviate part of this problem, will temporarily enable for interrupts and then disable. There are three occasions when the dispatcher enables for interruptions (enable windows):

1. When an enabled wait state is entered.
2. When an enabled problem state is entered to run a virtual machine.
3. When another part of the supervisor is entered via the unstacking of a CP request block.

On occasions 1 and 2, the dispatcher ignores the enable since the system will soon be enabled for interruptions. On occasion 3, if the dispatcher finds a CP request block to unstack, it first enables then disables for interruptions before unstacking the request.

Favored Execution Options

When the resources of the processor (and real storage) are being allocated, the dispatching and scheduling functions are implemented in such a manner that options exist which allow an installation to designate that certain virtual machines are to receive preferential treatment.

The favored execution options allow an installation to modify the algorithms described above and force the system to devote more of its resources to a given virtual machine than would ordinarily be the case. The options provided are:

1. The favored execution option.
2. The favored execution percentage.

The favored execution option means that the virtual machine so designated is never to be dropped from the active (in-queue) subset by the scheduler. When the virtual machine is executable, it is to be placed in the dispatchable list at its normal priority position. However, any active virtual machine represents either an explicit or implicit commitment of main storage. An explicit storage commitment can be specified by either the virtual=real option or the reserved page option. An implicit commitment exists if neither of these options are specified, and the scheduler recomputes the virtual machine's projected work-set at what it would normally have been at queue-drop time. Multiple virtual machines can have the basic favored execution option set. However, if their combined main storage requirements exceed the sytem's capacity, performance can suffer due to thrashing.

The basic favored execution option removes the primary source of elapsed time stretch-out in a loaded time-sharing environment. However, if the favored task is highly compute-bound and must compete for the processor with many other tasks of the same type, an installation can define the processor allocation to be made. In this case, the favored execution percentage option can be selected for the virtual machine. This option specifies that the selected virtual machine, in addition to remaining in queue, receives a given minimum percentage of the total processor time, if he can use it. The percentage is assured in the following manner:

1. The in-queue time slice is multiplied by the requested percentage and added to the virtual machine's current total processor time usage.

2. When the favored virtual machine, is executable, it is always placed at the top of the dispatchable list until it has obtained his guarantee.

3. If the virtual machine obtains its guarantee before the interval has elapsed, it is placed in the dispatchable list according to its caluculated dispatching priority.

4. In any case, at the end of the in-queue time slice, the guarantee is recomputed as in step 1 and the process repeated.

These options can impact the response time of interactive virtual machines and only one favored percentage virtual machine is allowed at any given time.

Dispatching and Scheduling Support Routines

Most of the routines in the CP nucleus are reenterable and multiple control program or virtual machine tasks can make use of one routine at the same time. However, there are certain areas where requests for a resource must be serialized (as in paging) or delayed while previous requests are serviced (as in requests to schedule I/O).

The CP Request Stack

The routine handling the request obtains a CPEXBLOK from free storage and stores the caller's registers in it; when the requested resource is free, the CPEXBLOK is stacked for the dispatcher via a call to the request stack manager (DMKSTK). The dispatcher unstacks the block and exits to the requesting routine the next time it is entered. I/O requests are stacked in the same manner, except that the stacking vehicle is the IOBLOK, and return is passed to the address specified in the interrupt return address (IOBIRA). In either case, it should be noted that the dispatcher always unstacks and gives control to any stacked IOBLOKs and CPEXBLOKs prior to dispatching a user. This guarantees that CP information needed by a virtual machine (such as page availability) is always as up to date as possible.

CP SPOOLING

The spooling support in CP performs three functions.

- Simulates the operation of the virtual unit record devices that are attached to each user's virtual machine configuration. The simulation is done in such a way that it appears to the program in the virtual machine that it is controlling a real unit record device. This support involves the interception and interpretation of virtual machine SIOs, the movement of data to and from the virtual machine's virtual storage space, and the reflection of the necessary interruption codes and ending conditions in PSWs, CSWs, and sense bytes. This support is provided by the virtual spooling executive.

- Operates the real unit record equipment, attached to the system, that transcribes virtual machine output spool files to the real printer or punch and input from the real card reader to DASD storage. This function is provided by the real spooling executive.

- Provides an interface among the virtual machines, the system operator, and the spooling system so that the location, format, priority and utilization of the systems spooling data and resources can be controlled.

SPOOL DATA AND FILE FORMAT

Data Format

The buffers that collect and write spool data are all one page (4096 bytes) in length, and contain the data to be transcribed and all CCWs necessary for operating the unit record devices that perform the transcription. The data is provided in the exact format required with no compression except that trailing blanks are suppressed. The first two doublewords of each buffer contain linkage information described below, followed by the data and CCWs, except for the first spool buffer which contains 3800-related information.

Each spool logical record (card or print line) is stored as one CCW that moves data (READ or WRITE), a TIC to the following CCW, and the full data record. Space is left at the end of each buffer so that a SENSE command can be inserted to force concurrent channel end and device end. For card punch channel programs there is an additional back chain field that points to the card previously punched so that error recovery for punch equipment checks can back up one card. The only exception to the format of READ/WRITE-TIC-Data is in buffers of files directed to the printer. In this case, immediate operation code CCWs (skips and spaces) are followed by the next CCW.

File Format

In addition to the data and CCWs contained in each spool buffer, the first two doublewords contain forward and backward links to the next and previous buffers in the file. This two-way linkage allows the file to be backspaced or restarted from any point at any time. Also, it means that if I/O errors are encountered while reading one buffer, the file is put in system hold status. If purged, all buffers except those in error are released. The two-way chain allows this control of the file while preventing fragmentation by allowing pages to be assigned and released individually regardless of their ownership.

The first spool buffer of an output spool file contains a special data record called the tag record. This record immediately follows the two doublewords containing the forward and backward buffer linkage pointers. The tag record allows VM/370 users to specify information to be associated with spool files that they generate. The information is entered via the CP TAG command, although the tag record is not considered a spool file data record and is not printed or punched as part of the spool file. However, the contents may be interrogated via the CP TAG QUERY command.

The format of the tag record is a NOP CCW, followed by a TIC to the next CCW and a 136-byte data field. To differentiate the tag record from an immediate NOP CCW (no TIC-data sequence) independently of the command code, the "skip" bit (bit 35) in the CCW has the following convention:

Bit 35 = 0 for NOP CCW, TIC, data (tag record)

= 1 for NOP CCW (immediate NOP command)

Each spool file in the system is controlled by a spool file control block (SFBLOK) that is resident in storage. While the file is open, these blocks are chained from the devices (either real or virtual) that are processing the file, and from device type file anchors after the file is closed. There is one file chain each for printer, reader, and punch files. Each SFBLOK contains information about the file that describes its owner and originator (these can be different for transferred files), the filename and filetype, and the class and number of copies for output files. All of these attributes can be examined and most can be changed by the file's owner or the system operator. The SFBLOK also contains information such as the starting and ending buffer addresses for the file, the record size, certain file status flags, etc.

SPOOL BUFFER MANAGEMENT

Real/Virtual Storage Management

Buffers that temporarily store spool data on its way between DASD secondary storage and the user's virtual machine are allocated from a pool of virtual storage space that belongs to CP. The size of this pool varies with the real storage available to VM/370 (the storage specified at system generation or actual real storage, whichever is less). Allocation is as follows:

Storage Size Available	Virtual Buffers Allocated
384K to 655,360 bytes	128
655,361 bytes to 1.1 megabytes	320
1.1 megabytes to 3 megabytes	640
over 3 megabytes	1280

Virtual storage buffers are allocated in 1-page increments by DMKPGT at the time the spool file is opened for either input or output. If no virtual storage space is available, the virtual machine is terminated with a PGT008 abend. This places limits on the number of concurrent spooling operations permitted by the system because spooling operates as a high-priority task.

Real storage is not allocated for a spooling buffer until a virtual machine actually issues a SIO that attempts to transfer data between the buffer and the user's virtual storage space. At this time, a page of real storage is allocated to the buffer via the real storage paging manager. The buffer is locked in main storage (that is, is unavailable to be paged out) only for the amount of time necessary to transfer the data. After the data transfer is complete, the buffer is treated as a normal page of virtual storage, and can be selected to be paged out. This ensures that low-usage spool files do not have buffers in real storage, while the buffers for high-usage files should remain resident. (Two spool file buffers are maintained for a 3800 printer.) The location of the spool buffer in real storage is transparent to the virtual spooling executive, because all references to the data therein are accomplished through the DAT feature of the processor.

DASD Space Allocation

While a spool buffer is inactive, it resides in real storage or on the paging device. After it has been filled with data from the virtual machine or a real input reader, it is written to a page of secondary DASD storage. The allocation of pages on the spooling disk(s) is managed by DMKPGT, which handles requests for both pages of virtual storage and semipermanent spool file residence. DMKPGT maintains separate allocation block chains for virtual storage and spooling pages. Each block contains control information and a bit map that allocates pages on a single cylinder. If none of the cylinders allocated have any available pages, DMKPGT enters its cylinder allocation routine.

DMKPGT attempts to even out the spooling and paging I/O load by allocating cylinders across channels and devices. To minimize seek times on a given device, cylinders are allocated as close to the relative center of the spooling or paging area as possible.

Paging Device Support: All actual I/O for the page buffers on any device is controlled by the paging I/O executive DMKPAGIO.

VIRTUAL SPOOLING MANAGER (DMKVSP)

The two functions of the virtual spooling manager are (1) to simulate the operation of all spooled unit-record devices attached to the user's virtual machine, and (2) to read and write the spool files associated with those devices. The following virtual devices are supported for spooling, with the exceptions noted:

- IBM 2540 Card Reader/Punch, except for punch feed read and column binary
- IBM 3203 Printer Model 4 and Model 5 (132 positions)
- IBM 1403 Printer Models 2 and N1 (132 positions)
- IBM 3211 Printer (150 print positions)
- IBM 3505 Card Reader (except for mark senses reading)
- IBM 3525 Punch (except for the card read, print, and data protect features).

The following consoles are supported for spooling when entered into the directory as the virtual system console:

- IBM 1052 Printer-Keyboard, Model 7 (via the 2150 Console)
- IBM 3210 Console Printer-Keyboard, Models 1 and 2
- IBM 3215 Console Printer-Keyboard, Model 1

All virtual printers must have the universal character set feature. No checking is done on the spooled printer data. However, any UCS buffer commands issued by the virtual machine (load UCS buffer, block data checks, etc.) are ignored. It is up to the user and the installation to ensure that the output is directed to the proper real printer via use of the output CLASS feature described below. For the 3211 or 3203 printer, forms control buffer (FCB) commands are accepted

and simulated by means of a virtual FCB maintained by the executive. The use of the virtual FCB is the only way to simulate end-of-form conditions reflected by the detection of a channel 9 or 12 punch. When the spooled file is directed to a real 3211, 3203, or 1403, the operator is responsible for loading the FCB or mounting the proper carriage tape.

If any of the unsupported unit record features are required, the real device must be attached directly to the user's virtual machine. Thus, a 3505 reader could be a spooling input reader, but attached directly to a batch virtual machine when it is necessary to read mark sense cards.

Output File Processing

DMKVSP receives control from the virtual I/O executive, DMKVIO, when the user's machine issues a SIO to a spooled unit record device. DMKVIO does not pass control until it has been determined that the device is available (that is, it is not busy and has no interruptions pending). DMKVSP first determines if the device is currently processing a file. If it is, processing continues. If this is the first command issued by the given device, a new output file must be opened. An open subroutine is called to build the control blocks necessary to manage the file and to obtain virtual storage and DASD buffer space. Control is then returned to DMKVSP.

Before the first record of an output spool file is written, DMKVSP writes a tag record (NOP CCW, TIC, data sequence) and initializes the 136-byte data area to blanks. It then sets the spool buffer displacement pointer to the first doubleword in the buffer beyond the tag record. DMSVSP then analyzes and interprets the channel program associated with the virtual machine's SIO. Each CCW is tested for validity of command, address, flags, alignment, protection, etc., and if the CCW is valid, the virtual machine's data is moved from his own virtual storage space to the buffer in the spooling virtual storage. When this buffer is full, it is written to a page of DASD secondary storage and a new buffer is obtained. The interpretation of the virtual machine's channel program continues until there are no more CCWs or until an error condition is detected that prohibits further processing. In either case, the device is marked as having the proper interruptions pending, a CSW is constructed, and DMKVSP exits to the main dispatcher. In contrast to nonspooled I/O, the virtual machine has remained in a pseudo-wait (IOWAIT) for the time it took to interpret the entire channel program.

The output file can be logically closed by the virtual machine either by issuing an invalid CCW command code, or by the CP CLOSE command. In either case, DMKSPL checks for tag record information
| and 3800-related information in the VSPXBLOK. (The VSPXBLOK, pointed to by the VDEVEXTN field of the VDEVBLOK for the output spool device, contains the tag information entered via the CP TAG command.) If tag data exists, the first spool buffer for the file is read in, the tag data is inserted in the tag record, and the buffer is rewritten to DASD storage. If no tag data exists, the tag record data field is left blank. The device is then cleared of pending interruptions, the file chains are completed, and the file is either queued for output on a real device of the proper type (printer or punch), or, if XFER is in effect, is queued for input to another virtual machine.

| The 3800-related information includes:

| CHARS - character arrangement table
| MODIFY - copy modification name
| FCB - file control block

| FLASH - flash count overlay use

| This information is contained in the VSPXBLOK for a virtual printer.
| When the file is closed, the information is contained in the first DASD
| buffer.

Input File Processing

Input file processing is similar to output file processing, except for the open and close functions, and the analysis of CCW commands and the direction of data movement. Many common routines are utilized to locate and verify CCWs, obtain buffer space, and to move the spooling data.

The difference in the open function is that instead of creating a new file, it is necessary to locate a reader file that already exists in the system. To do this, the open subroutine scans the SFBLOKs chained from the anchor, READERS, to find a file with an owner userid that matches that of the caller and is not in hold status. If a file is not found, a unit check or intervention required condition is reflected to the virtual machine; otherwise, its SFBLOK is chained to the control block for the reader and the channel program is interpreted in the same manner as for an output file.

After the input file is exhausted, a unit exception is reflected to the user machine, unless the user has requested either continuous spooling or that an EOF not be reflected. With continuous spooling, the unit exception is not reflected until the last file for that virtual machine is processed. If NOEOF is specified, the simulation terminates with a unit check or intervention-required condition (similar to what happens if the EOF button on a real reader is not pushed).

In either case, the input file is then deleted from the system, unless the user has specifically requested that his input files be saved. If the file is saved, it can be re-read any number of times.

Virtual Console Spooling

Support of virtual console I/O for both the virtual machine and VM/370 is provided as an option for the VM/370 spooling capabilities. This support fulfills the following requirements:

- Provides hardcopy support for CMS Batch Facility virtual machines.
- Provides hardcopy support for display devices used as system or virtual machine consoles.
- Allows disconnected virtual machines to spool virtual console output, CP commands and system resources to disk instead of losing the output.
- Improves the performance of virtual machines that currently produce a large amount of console output.

Whenever a SIO is issued to a virtual machine console, the virtual console manager (DMKVCN) determines if the spooling option is active. If it is, control is passed to the virtual spooling manager at DMKVSPBP to insert the data into a spool file buffer. While console spooling utilizes, basically, the same code as printer spooling, the following exceptions are made:

- A skip to channel 1 CCW is inserted after every 60 lines of output.
- The operator's virtual console spool buffer is written out after every 16 lines of output.
- The virtual spool buffer is written out to the allocated spool device when the first CCW is placed in that virtual buffer. The linkage area of the virtual spool buffer takes the form of a CLOSE file to allow checkpoint (DMKCKP) to recover the active spool file in the event of a shutdown because of system failure. If data in the virtual buffer has not yet been written to the spool device, it will not be recovered.

  To maintain a pseudo closed file status for console spool files, DMKSPL now assigns spool identifications to all output spool files where they are first queued.

  A virtual system reset, device reset, or IPL does not close the virtual console spool file. The LOGOFF, FORCE, or DETACH of virtual console commands does close the virtual console spool file. The SHUTDOWN command does close the operator's console spool file. If the SHUTDOWN command is issued by a Class A user other than the operator, the console spool file for both the user and operator is closed.

The inclusion of the spool file tag record in a virtual console spool file is processed by DMKVSP and DMKSPL as described for printer spool files in "Output File Processing" under "Virtual Spooling Manager."

REAL SPOOLING MANAGER (DMKRSP)

The real spooling manager operates the real unit record devices that are attached to the system and that are used to transcribe input data into reader spool files and user output spool files onto the real printers and punches. The executive optimizes the use of main storage and the processor rather than running the system unit record devices at their rated speeds. DASD input files are not double-buffered except for a 3800 printer, and under periods of peak load, input and output devices tend to run in bursts. However, command chaining is used for all unit record channel programs so that the devices are running at their maximum speed with a minimum of interruptions.

Output File Processing

Both the input and output operations of DMKRSP are interruption driven. Thus, DMKRSP does not process unless an internally or externally generated not-ready to ready device end interruption occurs. External interruptions are generated by the hardware in the normal manner, while internal, "pseudo interruptions," are generated by the software when an output file has been queued on the real printer or punch file chain, or when the operator issues a START command to a drained device.

Upon receipt of the initial device end for a printer or punch, DMKRSP searches the appropriate file chain for the SFBLOK of a file whose class matches that of the device that was made ready. If FLASH is specified for a 3800 printer, the flash overlay name must also match. When the SFBLOK is located (provided the file is not in a hold status), it is

unchained from the output queue and chained to the real device block that services the file. A page of real main storage (two pages for a 3800 printer) is then obtained for use as a buffer, and the output separator routine (DMKSEP) is called to print output identifier pages. DMKTCS is then called to set up the 3800 for printing that file. When DMKSEP returns control to DMKRSP, the first buffer of the file is paged into real main storage, and the CCWs in the channel program that it contains are adjusted so that their data addresses correspond to the real addresses at which the data resides. The real SIO supervisor (DMKIOSQR) is then called to start the channel program, and DMKRSP exits to the dispatcher (DMKDSPCH) to await the interruption.

When the channel end/device end interruption for the completed buffer is unstacked to DMKRSP, the forward chain file link field locates the next buffer. This buffer is paged-in, and the process is repeated until the final buffer is processed. At this point, the number of copies requested for the file is decremented. If the number of copies is 0, processing is terminated and the file is deleted from the system; otherwise, the process is repeated as many times as necessary. For a 3800 printer, double buffering is maintained so that the second buffer is filled while the first buffer is being printed.

When file processing is complete, a scan of the appropriate output queue is again made, and if a file is found it is processed. If the queue is empty, or if a file with a matching class is not found, an exit is taken to DMKDSPCH to wait for another ready interruption. If a 3800 device is used, the file is placed on the 3800 delayes purge queue. If this queue reaches maximum size, the oldest file in the queue is deleted from the system.

Output file processing can be modified by either the system operator, by a spooling support command or as a result of system errors. The operator commands allow a given file to be backspaced or restarted, and the files of individual users or the whole system to be held and released for output. I/O errors also affect the spooling system, and a description of how they are processed is in the section "Spool File Error Recovery."

Input File Processing

Reader file processing is initiated by the receipt of a device end interruption from a spooling card reader. No explicit operator command is required to start the processing of an input file. When the device end is unstacked to DMKRSP, an open subroutine is called to build the necessary control blocks and to obtain the virtual, real, and DASD buffer space required for the file. A channel program to read 41 cards is built in the buffer, and DMKIOSQR is called to start the reader.

When the interruption for the first buffer is unstacked, the first card is checked for its validity as a userid card. The minimum information that this card must contain is the userid of the owner of the input file. It may appear anywhere on the card, with the restriction that it must be the first information punched. Optional information on the userid card can include a filename and type and/or the class of the virtual card reader to which the file is to be directed. If the userid is valid, the file processing continues; otherwise, the operator receives an error message and processing is terminated.

After each file buffer is read, it is written onto disk by the paging I/O routines in the same way that virtual output files are handled. When a unit exception signaling physical end of file is received from

the reader, the file is closed by writing the final buffer to disk and completing and queuing the SFBLOK to the reader's file chain. If the owner of the file is currently logged on, he is given a message indicating that a file has been read and if he has an available card reader, it is posted with a device end interruption. An available reader is one of the correct class which is ready, is not busy, has no active file, and has no pending interruptions.

Accounting Card Processing

Various routines in CP accumulate, format, and punch account cards that contain system usage information for certain users. These routines format the information into an 80-column card image preceded by a punch CCW and call DMKACOAQ to queue the card for real output. DMKACOAQ calls DMKACOPU to punch the card on a real punch, if one is available; otherwise, the card is queued in main storage until a punch is free. When a punch finishes processing its last file, a test is made to see if any accounting cards have been queued. If they have, DMKACOPU is called to process them.

In addition to the cards generated by CP to account for a virtual machine's use of system resources, the user may request cards to be punched in order to account for the use of virtual machine resources by jobs running under his userid. In order to do so, the user must have the account option (ACCT) entered into the directory.

To punch an accounting card, the user must issue a code X'004C' DIAGNOSE instruction with a pointer to either a parameter list containing user-specified "charge to" information, or a data area containing up to 70 bytes of user-specified information to be punched into the accounting card. DMKHVC validates the instruction operands, builds an account buffer (ACNTBLOK), and DMKACOQU is called to queue the card for real output. For additional information about this user option, see "DIAGNOSE Interface (DMKHVC)" under "Privileged Instructions."

When the user accounting option is being utilized, the user must keep in mind that each additional accounting record requested is occupying real storage space. Degradation of system performance occurs if available storage becomes filled with accounting data.

SPOOLING COMMANDS

The spooling commands provide an interface between the user, the system operator, and the spooling system. There are three types of spooling commands:

- Those that affect virtual devices
- Those that affect real devices
- Those that affect spool files that are queued within the system

The commands that affect virtual devices are generally available to all system users, and a user can only affect the status of devices that are attached to his own virtual machine. Commands that affect the status of the real system's spooling devices can be used by the system operator only. Commands that affect closed spool files that are awaiting processing are generally available to all users, with some additional capabilities assigned to the system operator. For example, a

user may alter the characteristics only of those files that have an owner's userid that matches his own, whereas the system operator may change any spool file in the system.

File States and Attributes

Each spool file in the system has a number of attributes that are assigned to it, either explicitly or by default, at the time that it is created. These attributes and their values are as follows:

- Filename and filetype can be 24-character fields. Either or both can be replaced by a user-supplied value.
- Spoolid number is a system-assigned number between 1 and 9900. It is automatically assigned when the file is created (input) or closed (output), and is unique within the system. The file's owner, the device type, and the id number are specified. Usually, the userid defaults to the identification of the user issuing the given command. Because the identification number rather than the filename and filetype is an identifier, duplicate user-assigned names do not present an identification problem.
- The number of logical records (cards or print lines) in the file is an integer between 1 and 16 million. For printer files, the record count also includes any immediate operation code space or skip CCWs.
- The originating user is the identification of the file's creator, if the file has been internally transferred from the originator's printer or punch to the new owner's card reader.
- The number of copies requested for an output file is between 1 and 99. Unless altered by the user or operator, it defaults to 1.
- The device type is used by DIAGNOSE for a file transferred to a reader to determine the virtual type of output device.
- CHARS for 3800 printer
- FCB for 3800 printer
- MODIFY for 3800 printer
- FLASH for 3800 printer

In addition to those attributes, a file that is queued for real output or virtual input always has a class associated with it. A class is a single alphameric character from A through Z or from 0 to 9. It controls both the real or virtual device on which the file will be printed, punched, or read, and the relative priority and sequence of output on the device. While each file is assigned a single class, each real spooling output device can be assigned from one to four classes. The device then processes only files that have a class attribute that corresponds to one of its own, and processes these files in the order that its own classes are specified.

For example, if a printer is assigned the classes A, D, 2, it processes any printer file with a class of A before it searches the printer output queue for a file with class D. All class D files are printed before class 2 files.

The output class for a file is assigned at the time the file is created and is the class that is associated with the virtual device that created it. While each real spooling device can have up to four

classes, each virtual spooling device can have only one. When a user logs onto to the system, the class associated with a device is the one defined in his directory entry for that device. However, he can alter this class at any time by the SPOOL command. As files are created and closed by a device, they take on the device's output class.

After they are closed and are awaiting output, their class can be changed by a CHANGE command issued either by the file's owner or the system operator. The system operator can alter the system generated output class(es) of a real output device by the START command.

Output files transferred to a user's virtual reader can also be controlled by class. If the receiving user has several readers, the input to each can be limited to files of a certain class. In addition, the ORDER command allows sequencing of input files by class as well as spoolid number.

Output priorities can also be managed by altering the hold status of a file. Individual users can alter the hold status with the CHANGE command, while the system operator can change (hold or free) the files of specific individual users.

| SPOOL and CHANGE commands can be used to modify the CHARS, FCB, MODIFY,
| and FLASH attributes of a file or a virtual printer.

Virtual Device Spooling Commands

These commands affect the status of a user's virtual spooling devices:

Command	Meaning
CLOSE	Terminates spooling operations on a specified device. It clears the device of any pending interrupt conditions, and for output files, updates the tag record, completes and queues the file for real output. Optional operands allow the user to specify a filename and filetype, and to override for the given file any standard CLASS, HOLD/NOHOLD or COPY operands set into the output device by the SPOOL command.
SPOOL	Establishes the file attributes that apply to files created on, or read by, the given device. It establishes the class that will be in effect, whether: files are to be automatically held, input files are to be saved or purged after reading, and output files are to be directed to the real system printers and punches or are to be transferred to a user's virtual reader. The SPOOL command also specifies 3800 attributes.

Real Device Spooling Commands

The operator can use these commands to control the activity of the real spooling devices:

Command	Meaning
BACKSPAC	Backspaces an active spooling device for either a specified number of pages (printers only) or to the beginning of the file (printers or punches).
DRAIN	Stops the operation of a specified output or input device after it has finished processing the file on which it is currently working. A printer must be drained prior to the issuance of the LOADBUF command. Unit record devices are normally drained prior to system shutdown.

START Restart a device after it has been drained. Options allow the operator to specify the spooling output class for the output device and output separator records. For a 3800 printer, the IMAGE CHAR, FCB and PURGE options may also be specified.

FLUSH Immediately halts the output on the specified device and either flushes that copy of the file from the system, or puts it into the system hold status for future processing.

REPEAT Supplements the number of copies requested by the user for the file when it was created. The operator can specify a number from 1 to 99 that is added to the number specified by the user.

LOADBUF Loads the universal character set buffer of the FCB of the specified printer with the specified image. If requested, the system verifies the loading by printing its contents on the affected printer.

SPACE Forces the output on the specified printer to be single spaced, regardless of the skipping or spacing commands specified by the file's creator.

Spool File Management Commands: The spooling commands alter the attributes and status of closed spool files that are queued and awaiting processing. When a command applies to an individual file, the device type (RDR, PUN, PRT) and the spoolid number must be provided to identify the file. In most commands requiring a spoolid, the keyword CLASS followed by a valid spool class or the keyword ALL are acceptable substitutes for the spoolid number. This causes the command to be executed for all files of the given class or device type. The userid is the identification of the user issuing the command, except that the system operator must explicitly supply the identification of the user whose files he wishes to affect or he must specify the keyword SYSTEM, which gives access to all files (valid for CHANGE, PURGE, ORDER, and TRANSFER commands also).

Command Meaning

CHANGE Changes the filename and filetype, the number of copies, and the class of the specified file. The CHANGE command also specifies 3800 attributes. Any of the above attributes of a file can be determined via the QUERY command.

HOLD Places, via the system operator, the specified file in a hold status. The file is not printed or punched is released by the system operator. The operator can hold any user files by device type.

FREE Opposite of the HOLD command. Allows a file or group of files that were previously held to become available for processing. However, the user cannot reset a hold that was set by the operator with the HOLD command.

PURGE Removes unwanted spool files from the system before they are printed or punched.

ORDER Reorders the input files in a virtual card reader. It can order files by identification number, by class, or by any combination of the two.

TRANSFER Transfers a virtual reader to another user's virtual reader without any processing. The TRANSFER command causes a changing in the owning userid field in the file's SFBLOK.

Unit Record I/O Errors

I/O errors on real spooling unit record devices are handled by a transient routine that is called by DMKIOS after it has sensed the unit check associated with the error on a spooling device. If appropriate, a restart CAW is calculated and DMKIOS is requested to retry the operation, in some cases waiting for a device end that signals that the failing device has been made ready after manual corrective measures have been taken. If, after retrying the operation, the error is unrecoverable, DMKIOS is informed that a fatal error has occurred. DMKIOS then unstacks the interruption, flagged as a fatal error, and passes control to real spooling executive. The routines that handle unstacked interruptions in real spooling execute only module operations that have been completed correctly or those that are fatal errors. If a fatal error is unstacked, the recovery mechanism depends on the operation in progress.

For fatal reader errors, processing of the current file is terminated and any portion of the file that has been read and stored on disk is purged. The owner of the file is not informed of the presence of a fractional part of the file in the system.

For fatal printer or punch errors, the SFBLOK for the partially completed file is re-queued to the appropriate output list and processing can be resumed by another available printer or punch, or can be deferred until the failing device is repaired.

In any case, the failing device is marked logically offline, and no attempt is made by the system to use it until the operator varies it back online via the VARY command.

If an invalid load module is specified for a 3800 printer (refer to DIAGNOSE code X'74'), the file involved is held or purged, and the printer queue is searched for the next file to print. In addition, the user and operator are sent a message (DMKRSE241E), describing the action.

DASD Errors During Spooling

DASD I/O errors for page writes are transparent to the user. A new page for the buffer is assigned, the file linkage pointers are adjusted, and the buffer is rewritten. The failing page is not de-allocated and no subsequent request for page space is granted access to the failing page. If an unrecoverable error is encountered while reading a page, processing depends on the routine that is reading the file. If the processing is being done for a virtual reader, the user is informed of the error and a unit check/intervention required condition is reflected to the reader. If the processing is being done for a real printer or punch, the failing buffer is put into the system hold status, and processing continues with the next file. In either case, the DASD page is not de-allocated and it is not available for the use of other tasks.

DASD Spool Space Exhausted

If the space allocated for paging and spooling on the system's DASD volumes is exhausted and more is requested by a virtual spooling function, the user receives a message and a unit check intervention required condition is reflected to the virtual output device that is requesting the space, the output file is automatically closed and it is available for future processing. The user can clear the unit check and periodically retry the operation which will start when space is free or completely restart later from the beginning of the job. If the task requesting the space is the real spooling reader task, the operator receives an error message and the partially complete file is purged. Any time the spooling space is exhausted, the operator is warned by a console message and alarm. However, the system attempts to continue normal operation.

RECOVERY FROM SYSTEM FAILURE

Should the system suffer an abnormal termination, CP attempts to perform a warm start. Spool file and device data, as well as other system information is copied from real storage to warm start cylinders on DASD storage. When the system is reinitialized, the spool data and other system data is retrieved from the warm start cylinders and operation continues.

If the warm start data in real storage was damaged by the abnormal termination, the warm start procedure recognizes the situation and notifies the operator that a warm start cannot be performed. Another recovery method would be to attempt a checkpoint start.

The spool file recovery routines (DMKCKS) dynamically checkpoint on DASD storage; the status of all open reader files, the status of all closed output files, real spooling device data, and system hold queue information. This information is stored on checkpoint cylinders that are allocated, along with warm start cylinders, at system generation.

When a checkpoint (CKPT) start is requested, spool file and spooling device information is retrieved from the checkpoint cylinders. Spool file blocks are chained to their appropriate reader, printer or punch chains; record allocation blocks are reconstructed; spooling device status is restored; and, system hold queues are chained to the proper devices. System operation then continues.

If the checkpoint start procedure encounters I/O errors or invalid DASD data on the checkpoint cylinders, the operator is notified. The FORCE option of the checkpoint start performs all the checkpoint start functions except that, invalid or unreadable files are bypassed. While this is at best a partial recovery, the only other alternative is a cold (COLD) start, where all spool file data is lost.

RECOVERY MANAGEMENT SUPPORT (RMS)

The machine check handler (MCH) minimizes lost computing time caused by machine malfunction. MCH does this by attempting to correct the malfunction immediately, and by producing machine check records and messages to assist the service representatives in determining the cause of the problem.

The channel check handler (CCH) aids the I/O supervisor (DMKIOS) to recover from channel errors. CCH provides the device-dependent error recovery programs (ERPs) with the information needed to retry a channel operation that has failed.

This support is standard and model-independent on the external level (from the user's point of view there are no considerations, at system generation time, for model dependencies).

SYSTEM INITIALIZATION FOR RMS

DMKCPI calls DMKIOEFL to initialize the error recording at cold start and warm start. DMKIOEFL gives control to DMKIOG to initialize the MCH area. A store CPU ID (STIDP) instruction is performed to determine if VM/370 is running in a virtual machine environment, or running standalone on the real machine. If VM/370 is running in a virtual machine, the version code is set to X'FF' by DMKPRV. If the version code returned is X'FF', the RMS functions are not initialized beyond setting the wait bit on in the machine check new PSW (virtual). This occurs because machine check interruptions are not reflected to any virtual machine. VM/370, running on the real machine, determines whether the virtual machine should be terminated.

If the version code is not X'FF', DMKIOG determines what channels are online by performing a Store Channel ID (STIDC) instruction and saves the channel type for each channel that is online. The maximum machine check extended logout length (MCEL) indicated by the Store CPU ID (STIDP) instruction is added to the length of the MCH record header, fixed logout length and damage assessment data field. DMKIOG then calls DMKFRE to obtain the necessary storage to be allocated for the MCH record area (MCRECORD), the CP execution block (CPEXBLOK), MCHAREA, and MCEL. The address of MCHAREA is put in the PSA (AMCHAREA). Pointers to MCRECORD and the CPEXBLOK and put in MCHAREA. DMKIOG puts the address of MCEL in control register 15. DMKIOG obtains the storage for the I/O extended logout area and initializes the logout area and the ECSW to ones. The I/O extended logout pointer is saved at location 172 and control register 15 is initialized with the address of the extended logout area. The length of the CCH record and the online channel types are saved in DMKCCH. It should be noted that the ability of a CPU to produce an extended logout or I/O extended logout and the length of the logouts are both model- and channel-dependent. If VM/370 is being initialized on a Model 165 II or 168, the 2860, 2870, and 2880 standalone channel modules are loaded and locked by the paging supervisor and the pointers are saved in DMKCCH. If VM/370 is being initialized on any other model, the integrated channel support is assumed; this support is part of the channel control subroutine of DMKCCH. Before returning to DMKIOE, the VM/370 error recording cylinders are initialized. DMKIOE passes control back to DMKCPI and control register 14 is initialized with the proper mask to record machine checks.

OVERVIEW OF MACHINE CHECK HANDLER

A machine malfunction can originate from the processor, real storage or control storage. When any of these fails to work properly, the processor attempts to correct the malfunction.

When the malfunction is corrected, the machine check handler (MCH) is notified by a machine check interruption and the processor logs out fields of information in real storage, detailing the cause and nature of the error. The model-independent data is stored in the fixed logout area

and the model-dependent data is stored in the extended logout area. The machine check handler uses these fields to analyze the error, format an error record, and write the record out on the error recording cylinder of SYSRES.

If the machine fails to recover from the malfunction through its own recovery facilities, the machine check handler is notified by a machine check interruption. An interruption code, noting that the recovery attempt was unsuccessful, is inserted in the fixed logout area. The machine check handler then analyzes the data and attempts to keep the system as fully operational as possible.

Recovery from machine malfunctions can be divided into the following categories: functional recovery, system recovery, operator-initiated restart, and system repair. These levels of error recovery are discussed in their order of acceptability, functional recovery being most acceptable and system repair being least acceptable:

FUNCTIONAL RECOVERY: Functional recovery is recovery from a machine check without adverse effect on the system or the interrupted user. This type of recovery can be made by processor retry, the ECC facility, or the machine check handler. Processor retry and ECC error correcting facilities are discussed separately in this section because they are significant in the total error recovery scheme. Functional recovery by MCH is made by correcting storage protect feature (SPF) keys and intermittent errors in real storage.

SYSTEM RECOVERY: System recovery is attempted when functional recovery is impossible. System recovery is the continuation of system operations at the expense of the interrupted user, whose virtual machine operation is terminated. System recovery can only take place if the user in question is not critical to continued system operation. An error in a system routine that is considered to be critical to system operation precludes functional recovery and would require logout and a system dump followed by reloading the system.

OPERATOR-INITIATED RESTART: When the errors may have caused a loss of supervisor or system integrity, the system is put into a disabled wait state. The operator is instructed to run the standalone error recovery (SEREP) program and then manually restart the system.

SYSTEM REPAIR: System repair is recovery that requires the services of maintenance personnel and takes place at the discretion of the operator. Usually, the operator has tried to recover by system-supported restart one or more times with no success.

SYSTEM/370 RECOVERY FEATURES

The operation of the Machine Check Handler depends on certain automatic recovery actions taken by the hardware and on logout information given to it by the hardware.

Processor Retry

Processor errors are automatically retried by microprogram routines. These routines save source data before it is altered by the operation. When the error is detected, a microprogram returns the processor to the beginning of the operation, or to a point where the operation was executing correctly, and the operation is repeated. After several unsuccessful retries, the error is considered permanent.

ECC Validity Checking

ECC checks the validity of data from real and control storage, automatically correcting single-bit errors. It also detects multiple-bit errors but does not correct them. Data enters and leaves storage through a storage adapter unit. This unit checks each doubleword for correct parity in each byte. If a single-bit error is detected, it is corrected. The corrected doubleword is then sent back into real or control storage and on to the processor. When a multiple-bit error is detected, a machine check interruption occurs, and the error location is placed in the fixed logout area. MCH gains control and attempts to recover from the error.

Control Registers

Two control registers are used by MCH for loading and storing control information (see Figure 21). Control register 14 contains mask bits which specify whether certain conditions can cause machine check interruptions and mask bits which control conditions under which an extended logout can occur. Control register 15 contains the address of the extended logout area.

Word	Bits	Name of Field	Associated with
14	0	Check-stop control	Mch-Chk handling
14	1	Synchronous MCEL control	Mch-Chk handling
14	2	I/O extended logout control	Chan-Chk handling
14	4	Recovery report mask	Mch-Chk handling
14	5	Degradation report mask	Mch-Chk handling
14	6	External damage report mask	Mch-Chk handling
14	7	Warning mask	Mch-Chk handling
14	8	Asynchronous MCEL control	Mch-Chk handling
14	9	Asynchronous fixed log control	Mch-Chk handling
15	8-28	MCEL address	Mch-Chk handling

Figure 21. RMS Control Register Assignments

Machine Check Handler Subroutines

VM/370 Machine Check Handler module (DMKMCH) consists of the following functions:

- Initial analysis subroutine
- Main storage analysis subroutine
- SPF analysis subroutine
- Recovery facility mode switching
- Operator communication subroutine
- Virtual user termination subroutine
- Soft recording subroutine
- Buffer error subroutine
- Termination subroutine

Initial Analysis Subroutine

The initial analysis subroutine of DMKMCH receives control by a machine check interruption. To minimize the possibility of losing logout information by recursive machine check interruptions, the machine check new PSW gives control to DMKMCH with the system disabled for further interruptions. There is always a danger that a machine malfunction may occur immediately after DMKMCH is entered and the system is disabled for interruption. Disabling all interruptions is only a temporary measure to give the initial analysis subroutine time to make the following emergency provisions:

- It disables for soft machine check interruptions. Soft recording is not enabled until the error is recorded.
- It saves the contents of the fixed and extended logout areas in the machine check record.
- It alters the machine check new PSW to point to the term subroutine. The term subroutine handles second machine check errors.
- It enables the machine for hard machine check interruption.
- If a virtual user was running when the interruption occurred, the running status (GPRs, FPRs, PSW, M.C. old PSW, CRs, etc.) is saved in the user's VMBLOK.
- It initially examines the machine check data for the following error types:

  MCIC=ZERO
  PSW invalid
  System damage
  Timing facilities damage
  Channel inoperative on 3031/3032/3033 processor

  The occurrence of any of these errors is considered uncorrectable by DMKMCH; the primary system operator is informed, the error is formatted and recorded, and the system enters a wait state, code 001 or 013.

- If the instruction processing damage bit is on, it tests for the following types of malfunctions:

  -- Multiple-Bit Error in Main Storage -- Control is given to the main storage analysis subroutine.

  -- SPF Key Error -- Control is given to the SPF analysis subroutine.

  -- Retry failed -- If the processor was in supervisor state the error is considered uncorrectable and the VM/370 system is terminated. If the processor was in problem state, the virtual machine is reset or terminated and the system continues operation.

- If processor retry or ECC was successful on a soft error, control is given to the soft recording subroutine to format the record, write it out on the error recording cylinder, and update the count of soft error occurrences.
- If external damage was reported, control is given to the soft recording subroutine to format the record and write it out on the error recording cylinder.

Main Storage Analysis Subroutine

The main storage analysis subroutine is given control when the machine check interruption was caused by a multiple-bit storage error. An initial function points the machine check new PSW to an internal subroutine to indicate a solid machine check, in case a machine check interruption occurs while exercising main storage.

Damaged storage areas associated with any portion of the CP nucleus itself cannot be refreshed; multiple-bit storage errors in CP cause the VM/370 system to be terminated. An automatic restart reinitializes VM/370.

If the damage is not in the CP nucleus, main storage is exercised to determine if the failure is solid or intermittent. Multiple-bit ECC storage errors on a 3031, 3032, or 3033 processor are always treated as solid errors. If the failure is solid, the 4K page frame is marked unavailable for use by the system. If the failure is intermittent, the page frame is marked invalid. The change bits associated with the damaged page frame are checked to determine if the page had been altered, by the virtual machine. If no alteration had occurred, VM/370 assigns a new page frame to the virtual machine and a backup copy of the page is brought into storage the next time the page is referenced. If the page had been altered VM/370 resets or terminates the virtual machine, clears its virtual storage, and sends an appropriate message to the user. Normal system operation continues for all other users.

Storage Protect Feature (SPF) Analysis Subroutine

The SPF analysis subroutine is given control when the machine check interruption was caused by an SPF error. An initial function points the machine check new PSW to an internal subroutine if a machine check interrruption occurs during testing and validation. The SPF analysis routine then determines if the error was associated with a failure in virtual machine storage or in the storage associated with the control program.

An SPF error associated with VM/370 is a potentially catastrophic failure. Namely, VM/370 always runs with a PSW key of zero, which means that the SPF key in main storage is not checked for an out-of-parity condition. The SPF analysis subroutine exercises all 16 keys in the failing storage 2K page frame. If an SPF machine check occurs in exercising the 16 keys 5 times each, the error is considered solid and the operating system is terminated with a system shutdown. If an SPF machine check does not occur, the machine check is considered intermittent. The zero key is restored to the failing 2K page frame and this is transparent to the virtual machine.

If an SPF machine check occurs, which is associated with a virtual machine, the SPF analysis subroutine exercises all 16 keys in the failing storage 2K page frame. If an SPF machine check does not occur, the machine check is intermittent and the SWPTABLE for the page associated with the failing storage address is located. The storage key for the failing 2K storage page frame is retrieved from the SWPTABLE and the change and reference bits are set on in the storage key. The storage key is then stored into the affected failing storage 2K page frame. If an SPF machine check occurs in exercising the 16 keys 5 times each, then the machine check is considered solid and the following actions are taken. (1) The virtual machine is selectively reset or terminated by the virtual machine termination subroutine; (2) The 4K page frame associated with the failing address is removed as an

available system resource. This is accomplished by locating the CORTABLE for the defective page and altering the CORFPNT and CORPBPNT pointers to make the page unavailable to the system. The CORDISA bit in this CORTABLE is set on to identify the reason for the status of this page in a system dump.

Recovery Facility Mode Switching

The recovery facility mode switching subroutine (DMKMCHMS) allows the service representative to change the mode that processor retry and ECC recording are operating in. This subroutine receives control when a user with privilege class F issues some form of the SET command with the MODE operand. A check is initially made to determine if this is VM/370 running under VM/370. If this is the case, the request is ignored and control is returned to the calling routine. For the format and usage of the SET command with the MODE operand, refer to the VM/370 Operator's Guide.

Operator Communication Subroutine

The operator communciation subroutine is invoked when the integrity of the system has degraded to a point where automatic shutdown and reload of the system has been tried and was unsuccessful, or could not be attempted due to the severity of the hardware failure. A check is first made to determine if the system operator is logged on as a user, next a check is made to determine if the system operator is disconnected. If either of these checks is not affirmative a message cannot be issued directly to the system operator. A LPSW is performed to place the processor in a disabled wait state with a recognizable wait state code in the processor instruction counter.

Virtual User Termination Subroutine

The virtual machine termination subroutine selectively resets or terminates a virtual user whose operation has been interrupted by an uncorrectable machine check. First, the machine is marked nondispatchable to prevent the damaged machine from running before reset or termination is performed. The machine check record is formatted and DMKIOEMC is called to record the error. Then the user is notified by a call to DMKQCNWT that a machine check has occurred and that his operation is terminated. The primary system operator is notified of the virtual user termination by a message issued by a call to DMKQCNWT. If the virtual machine is running in the virtual=real area, DMKUSO is called to log the virtual machine off the system and to return the storage previously allocated to the virtual machine and to clear any outstanding virtual machine I/O requests. The HOLD option of LOGOFF is invoked to allow a user on a dial facility to retain the connection and thus permit LOGON without re-establishing the line connection. However, if the virtual machine is running in the virtual area, and DMKCFM is then called to put the virtual machine in console function mode, the user must re-initialize the system to commence operation.

Soft Recording Subroutine

The soft recording subroutine performs two basic functions:

- Formats a machine check record and calls DMKIOEMC to record the error on the error recording cylinder.
- Maintains the threshold for processor retry and ECC errors and switches from recording to quiet mode when the threshold value is exceeded. To accomplish this, a counter is maintained by DMKMCH for successful processor retry and corrected ECC events.

Processor Retry Recording Mode: Recording mode (bit 4 of control register 14 set to one) is the initialized state, and normal operating state of VM/370 for processor retry errors. Recording mode may also be entered by use of the CP SET command. When 12 soft machine checks have occurred, the soft recording subroutine switches the processor from recording mode to quiet mode. For the purpose of model-independent implementation this is accomplished by setting bit 4 of control register 14 to zero. Because in quiet mode no soft machine check interruptions occur, a switch from quiet mode to recording mode can be made by issuing the SET MODE RETRY|MAIN RECORD command. While in recording mode, corrected CPU RETRY|MAIN reports are formatted and recorded on the VM/370 error recording cylinder, but the primary systems operator is not informed of these occurrences.

Processor Retry Quiet Mode: Quiet mode (bit 4 of control register 14 set to 0) can be entered in one of two ways: (1) when 12 soft machine checks have occurred, or (2) when the SET MODE RETRY QUIET command is executed by a class F user. In this mode, both processor retry and ECC reporting are disabled. The processor remains in quiet mode until the next system IPL (warm start or cold start) occurs or a SET MODE RETRY|MAIN RECORD command is executed by a class F user. SET MODE MAIN is treated as invalid on a 3031, 3032, or 3033 processor.

ECC Recording Modes: To achieve model-independent support, RMS does not set a specific mode for ECC recording. The mode in which ECC recording is initialized depends upon the hardware design for each specific processor model. For the IBM System/370 Models 135, 135-3, 138, 145, 145-3, 148, 158, 168, 3031, 3032, and 3033, the hardware-initialized state (therefore the normal operational state for VM/370) is quiet mode. For the IBM System/370 Models 155 II and 165 II, the hardware initialized state (the normal operational state for VM/370) is record mode. An automatic restart incident due to a VM/370 failure does not reset the ECC recording mode in effect at the time of failure.

The change from record to quiet mode for ECC recording can be initiated in either of the following ways: (1) by issuing the SET MODE {MAIN|RETRY} QUIET command, or (2) automatically whenever 12 soft machine checks have occurred. For the purpose of model-independent implementation, this occurs by setting bit 4 of control register 14 to zero.

The change from quiet to record mode for ECC recording can be accomplished by use of the SET MODE MAIN RECORD command. This recording mode option is for use by maintenance personnel only. It should be noted that processor retry is placed in recording mode if it is not in that state when the SET MODE MAIN RECORD command is issued.

While in recording mode, corrected ECC reports are formatted and recorded on the error recording cylinder, but the primary systems operator is not informed of these incidents.

Buffer Error Subroutine

On processor models equipped with a high-speed buffer (155 II, 158, 165 II, 168, 3031, 3032, 3033) or a data lookaside table (DLAT) (165 II, 168, 3031, 3032, 3033) the deletion of buffer blocks because of hardware failure is reported via a degradation report machine check interruption. MCH enables itself for degradation report machine check interruptions at system initialization by setting bit 5 of control register 14 to 1. If a machine check interruption occurs that indicates high-speed buffer or DLAT damage, MCH formats the record and calls DMKIOEMC to record it on the error recording cylinder, informs the primary systems operator of the failure, and returns control to the system to continue normal operation.

Termination Subroutine

The termination subroutine is given control if a hard machine check interruption occurs while DMKMCH is in the process of handling a machine check interruption. Note that soft error reporting is disabled for the entire time that MCH is processing an error.

An analysis is performed of the machine check interruption code of the first error to determine if it was a soft error. If it was, the first error is recorded, the system status is restored and control is restored to the point where the first error occurred. If the first error was a hard error, the operator communication subroutine is given control to issue a message directly to the system operator, and to terminate CP operation.

OVERVIEW OF CHANNEL CHECK HANDLER

The channel check handler (CCH) aids the I/O supervisor in recovering from channel errors and informs the operator or service representative of the occurrence of channel errors.

CCH receives control from the I/O supervisor when a channel data check, channel control check, or interface control check occurs. CCH produces an I/O error block (IOERBLOK) for the error recovery program and a record to be written on the error recording cylinder for the system operator or service representative. The operator or service representative may obtain a copy of the record by using the CMS CPEREP command. A message about the channel error is issued to the system operator each time a record is written on the error recording cylinder.

When the I/O supervisor program detects a channel error during routine status examination following an SIO, TIO, HIO, or an I/O interruption, it passes control to the channel check handler (DMKCCH). DMKCCH analyzes the channel logout information and constructs an IOERBLOK and, if the error is a channel control or interface control check, an ECSW is constructed and placed in the IOERBLOK. The IOERBLOK provides information for the device-dependent error recovery procedures. DMKCCH also constructs a record to be recorded on the error recording cylinder. Normally, DMKCCH returns control to the I/O supervisor after constructing an IOERBLOK and a record. However, if DMKCCH determines that system integrity has been damaged (system reset or invalid unit address, etc.), then CP operation is terminated. CP termination causes DMKCCH to issue a message directly to the system operator and place the processor in a disabled wait state with a recognizable wait code in the processor instruction counter.

Normally, when DMKCCH returns control to the I/O supervisor, the error recovery program for the device which experienced the error is scheduled. When the ERP receives control, it prepares to retry the operation if analysis of the IOERBLOK indicates that retry is possible. Depending on the device type and error condition, the ERP either effects recovery or marks the event fatal and returns control to the I/O supervisor. The I/O supervisor calls the recording routine DMKIOE to record the channel error.

The primary system operator is notified of the failure, and DMKIOE returns control to the system and normal processing continues.

If the channel check is associated with an I/O event initiated by a SIO in a virtual machine, the logout is reflected to the virtual machine in one of two ways, depending upon whether the channel check occurred at SIO time or later in an interrupt. If it occurred at SIO time, then DMKVSI (or occasionally DMKVIO) calls upon DMKCCHRF to reflect the logout. If it occurred in an I/O interrupt, the dispatcher notices the channel check as it is reflecting the I/O interrupt to the virtual machine, and so, at that time, DMKDSP calls upon DMKCCHRF to reflect the logout.

CHANNEL CONTROL SUBROUTINE

Control is passed to the channel control subroutine of DMKCCH after a SIO with failing status stored, or an I/O interrupt because of a channel control check, interface control check, or channel data check.

If "logout pending" is indicated in the CSW, the CP termination flag is set. The existence of real device blocks (RCHBLOK, RCUBLOK, RDEVBLOK), for the failing device address, is determined by a call to DMKSCNRU and an indicator is set if they do exist. An indicator is also set if the IOBLOK for the failing device address exists. A call to DMKFREE obtains storage space for the channel check record and the channel control subroutine builds the record. If the indicators show that the real device blocks and the IOBLOK exist, a call to DMKFREE obtains storage space and the channel control subroutine builds the I/O error block (IOERBLOK); if these blocks do not exist, the IOERBLOK is not built. The IOERBLOK is used for two purposes:

1. The device-dependent error recording program (ERP) uses the IOERBLOK to attempt recovery on CP-initiated I/O events. If the I/O events that resulted in a channel check are associated with a virtual machine, the I/O fatal flag is set in the IOBLOK and the virtual machine is reset, cleared, and put into CP read status. The length and address of the channel check record is placed in the IOERBLOK and the IOERBLOK is chained off the IOBLOK.

2. DMKIOECC uses the IOERBLOK to record the channel check record on the error recording cylinder.

The channel control subroutine gives control to a channel-dependent error analysis routine to build or save the extended channel status word (ECSW). When the channel control subroutine regains control, eight active addresses are saved in the channel check record.

If the CP termination flag is set, the I/O extended logout data from the channel check record is restored to main storage for use by SEREP. If the system operator is both logged on as a user and connected to the system, a message (DMKCCH603W) is sent to him advising him of the channel error. A LPSW is then executed to place the processor in a disabled wait state with a wait state code of 002 in the processor instruction counter.

If the CP termination flag is not set, a check is made to determine if an IOERBLOK was built by the channel control subroutine.

If an IOERBLOK was not built, DMKIOECC is called to record the channel check record on the error recording cylinder. The system operator is then sent a message (DMKCCH601I or DMKCCH602I) informing him of the error and control is then returned to DMKIOS to continue system operation.

If an IOERBLOK was built, control is returned to DMKIOS, which calls the appropriate ERP. Whether or not recovery is successful, DMKIOS eventually calls DMKIOE to record the channel check record. DMKIOE examines the status of the in CSW error in the IOERBLOK to determine if it was a channel error; if so, it finds the length and pointer to the channel check record and records the error on the error recording cylinder. If this was not a channel error, DMKIOE continues normal processing.

INDIVIDUAL ROUTINES

A separate channel error analysis routine is provided for each type of channel for which DMKCCH can be used. The purpose of these routines and the channel control subroutine is to analyze the channel logout to determine the extent of damage and to create a sequence and termination code to be placed in the ECSW in the IOERBLOK. At system initialization, the correct model dependent channel recovery routine is loaded and the storage necessary to support the routine is allocated. The model-dependent error analysis subroutines and routines and their functions are as follows:

Integrated Channels (Models 135, 135-3, 138, 145, 145-3, 148, 155 II, 158, and 3031, 3032, and 3033 Processors)

Since all of these systems have integrated channels one common subroutine is used to handle all of these processor types. This subroutine:

- Indicates CP termination if the ECSW is not complete, the channel has been reset, the reset codes are invalid, or the I/O interface is inoperative.
- Moves the ECSW to the IOERBLOK
- Moves the hardware stored unit address and the I/O extended logout to the channel check record
- Sets the I/O extended logout area and ECSW area to ones
- Returns control to the channel control subroutine

2860 Channel (Models 165 II, 168)

The 2860 logout area is checked to determine if a complete logout exists; if not, CP termination is necessary.

A check is made in the logout area for validity of the CSW fields and bits are set in the channel check record's ECSW field to indicate bad fields.

The channel logout is then checked and sequence codes are set based on the presence of a channel control check, or an interface control check. If a channel control check is present, the codes set are determined through parity. The count determines if parity is good and sets a resultant condition code.

The logout area is examined to ensure that the unit address has valid parity and is the same address passed by DMKIOS. If so, the unit-address-valid bit in the ECSW is set. If the unit address is not valid, the unit-address-valid bit is reset to indicate the invalid condition.

The ECSW field in the channel check record is moved to the IOERBLOK, if one exists.

After completing the ECSW the 2680 routine moves the 2860 I/O extended logout into the channel check record, set the I/O extended logout area to ones, and returns to the channel control subroutine.

2870 Channel (Models 165 II, 168)

If the channel failed to log out completely, at least part of the logout area is all ones. If a fullword of ones is found, a CP termination condition exists.

A check is made in the logout area for valid CSW fields, and bits are set in the channel check record's ECSW field to indicate bad fields.

The termination and sequence codes are set depending on the presence of an interface control check or channel control check. If a channel control check is present, the codes set are determined through parity, count, and/or data transfer checks. For the 2870, parity can be determined directly from the channel logout.

The logout area is also examined to ensure valid parity in the unit address and to ensure that the address is the same as that passed to DMKCCH by DMKIOS. If so, the unit-address-valid bit in the ECSW is set.

The third word of the logout area is also analyzed for type II errors. If a type II error is found, a CP termination condition exists.

The ECSW field in the channel check record is moved to the IOERBLOK, if one exists.

Before returning to the channel control subroutine, the 2870 routine moves the 2870 I/O extended logout into the channel check record and sets the I/O extended logout area to ones.

2880 Channel (Models 165 II and 168)

This routine analyzes 9 words of the 28-word logout.

The 2880 analysis routine handles channel data checks, interface control checks, and channel control checks.

Termination code 3 (system reset) is not set in the ECSW because the 2880 channel does not issue system reset to the devices. Retry codes of 0 to 5 are possible.

Note: There are several catastrophic conditions under which the CP termination flag can be set, in the 2880 analysis routine. They are:

- The channel did not complete the logout.
- The CSW is not reliable.
- The unit address in the I/O interruption device address field is not correct.

Only a channel check record is needed if the channel has recognized an internal error and has recovered from it without any damage. No recovery action is necessary in these cases.

If the channel address in the I/O interruption device address field does not match the channel address in the logout, a CP termination condition exists.

If the channel was doing a scan and the unit control word had a parity check a CP termination condition exists. If there was no parity check, there was no damage during the scan and only a channel check record is required.

Depending on the sequence the channel has entered, the termination and sequence codes are set; command address, unit address, and unit status validity is determined; and the sequence code is set valid. The ECSW field in the channel check record is moved into the IOERBLOK, if one exists.

Before returning to the channel control subroutine, the 2880 routine moves the I/O extended logout into the channel check record and sets the I/O extended logout area to ones.

ERROR RECORDING INTERFACE FOR VIRTUAL MACHINES

The error recording interface provides a means of recording errors encountered by operating systems running in a virtual machine under VM/370. If the virtual operating system is VM/370, it must be the Release 2.0 version or later. An SVC 76 issued by a virtual machine is used to signal VM/370 that error recording is required. The SVC interruption handler in DMKPSA examines general registers 0 and 1 to determine if valid parameters have been passed. If valid parameters are not found, the SVC is reflected back to the virtual machine and no recording takes place. If valid parameters are passed, a pageable routine (DMKVER) processes the error record.

DMKVER validates the record passed by the virtual machine. If invalid conditions are found, no recording takes place. Control is returned to the SVC interruption routine in DMKPSA to reflect the SVC to the virtual machine as an SVC interruption. The action taken by the virtual machine is dependent on the operating system running in the virtual machine, not VM/370. If the record is valid, it is modified by changing virtual information to real. The actual recording is accomplished by using existing modules in DMKIOE and DMKIOF.

Control is then returned to the instruction following the SVC 76 rather than reflecting the SVC. This eliminates the duplication of error recording in VM/370 and the operating system in the virtual machine. If DMKVER determines that the recording represented a permanent I/O error, a message is sent to the primary system operator.

ERROR RECORDING AND RECOVERY

The error recording facility is made up of four modules. One module (DMKIOE) is resident and the other three (DMKIOC, DMKIOF, and DMKIOG) are pageable.

The error recording modules record temporary errors (statistical data recording) for CP generated I/O except for DASDs with a buffered log.

The error recording routines record: unit checks, statistical data counter overflow records, selected temporary DASD errors, machine checks, channel checks, and hardware environmental counter sense data on the error recording cylinders of the system resident device in a format suitable for subsequent processing by the CPEREP command (DMSIFC). The recorder asynchronously updates the statistical data counters for supported devices. The recorder also initializes the error recording cylinders at IPL if they are in an unrecognizable format.

When the recorder is entered from DMKIOS, it is entered at DMKIOERR. This entry is used for unit checks and channel data checks. A test is made of the failing CSW (located in the IOERBLOK) to see if the error was a channel error. If it was, control is passed to the routine for recording channel checks.

The IOERBLOK sense data, IOBLOK flags, and VMBLOK privilege class are examined to determine if the error should be recorded.

ERROR RECORD WRITING

After an error record is formatted, it is added to the error recording cylinder using DMKRPAGT and DMKRPAPT. The error recording cylinders have page-sized records (4096 bytes). Each page contains a header (8 bytes) which signifies: the cylinder and page number of the page (4 bytes), the next available space for recording within page (2 bytes), a page-in-use indicator (1 byte), and a flag byte. Each record within the page is recorded with a 4-byte prefix.

If an error record is too large to be added into a page, a new page is retrieved, updated with record, and placed back on the error recording cylinder with the paging routines.

From two to nine cylinders are used for error recording; errors are recorded in the order in which they occur. The cylinders that are used for error recording are specified by the installation or system programmer at system generation time. If the error recording cylinders become 90 percent full, a message is issued to the operator using DMKQCNWT to warn him of the condition. If the cylinders become full, another message is issued to inform the operator and recording is stopped.

On the 3031, 3032, and 3033 processors, frame records are read from the SRF device and written on the error recording cylinders during initialization if no records exist after a CPEREP CLEARF operation.

If a channel check error is to be recorded, the recorder is entered at DMKIOERR or DMKIOECC. The channel check handler determines the entry. A channel check error record is formatted.

A machine check enters at DMKIOEMC. Pointers are passed from the machine check handler in registers 6 and 7 to locate a buffer where the machine check record and length are saved. A machine check error record is recorded with the saved machine check logout and additional information. The machine check error record is written onto the error recording cylinder by using the paging routines.

Hardware environmental counter records are formed using routine DMKIOEEV. This routine is scheduled by DMKIOS after control is returned from the ERP. Sense data information is stored in the IOERBLOK by the ERP. The record formed is called a nonstandard record.

Clear and Format Recording Area

DMKIOEFM is called by DMSIFC (CPEREP command) via a DIAGNOSE instruction. DMKIOEFM is invoked to reset the specified error recording cylinders (if CLEAR, CLEARF, or ZERO=Y was specified). The clear is performed by resetting each page-header, space-available field. Pointers in storage are then updated to address the first available page on each of the error recording cylinders. Control is then returned to the calling routine. For details on the CPEREP command and EREP execution, refer to the VM/370 OLTSEP and Error Recording Guide and OS/VS EREP publications.

CLEARF on a 3031, 3032, or 3033 processor clears the cylinders, then causes the frame records to be read from the SRF device.

Find First Recording Cylinder at IPL

DMKIOEFL is called by DMKCPI to find the first available page that can be used for error recording. The paging routines, DMKRPAPT and DMKRPAGT, are used to read the error recording cylinders' pages (4096-byte records). As each page record is read, it is examined to see if this record is the last recorded. If so, a pointer in storage is saved so recording can continue on that page record. Control is then returned to the caller. If any error recording cylinder is in an unrecognizable format, the error recording area is automatically reformatted by CP.

DASD ERROR RECOVERY, ERP (DMKDAS)

Error recovery is attempted for CP-initiated I/O operations to its supported devices and for user-initiated operations to CP-supported devices that use a DIAGNOSE interface. The primary control blocks used for error recovery are the RDEVBLOK, the IOBLOK and the IOERBLOK. In addition, auxiliary storage is sometimes used for recovery channel programs and sense buffers.

The initial error is first detected by the I/O interruption handler which performs a SENSE operation if a unit check occurs. Unit check errors are then passed to an appropriate ERP. If a channel check is encountered, the channel check interruption handler determines whether

or not retry is possible and passes control to an ERP through the I/O interruption handler. DASD errors are processed as described below.

Channel Errors

- I/O interface inoperative on a 3031, 3032, or 3033 processor is reflected to the virtual machine if the channel is dedicated. Otherwise, a wait state X'0002' occurs.
- Channel control check is treated as seek check. It is retried 10 times.
- Interface control check is treated as seek check. It is retried 10 times.
- Channel data check is treated as data check. It is retried 10 times.

Unit Check Errors

Equipment check: Retry the operation 10 times for 3330, 3340, 3350, and 2305 devices; twice for the 2314 and 2319.

No record found and missing address marks: Recalibrate and retry the channel program 10 times (2314/2319).

No record found: Execute a READ HOME ADDRESS and check home address against seek address. If they are the same, consider the error permanent. If they are not equal recalibrate and retry the channel program 10 times (2314/2319). For other devices, return to caller.

Seek check: Retry the operation 10 times except that 3330/3350 seek checks are retried by hardware.

Intervention required: Issue a message to console and wait for solicited device end. This procedure is repeated once.

Bus out check: One retry of the operation.

Data checks: For 2314/2319 retry the operation 256 times, with a recalibrate being executed every 16th time. For the 2305/3340, retry the operation 10 times. For the 3330/3350, the operation is retried by hardware.

Overrun: Retry the operation 10 times.

Missing address marker: Retry the operation 10 times.

Command reject: The command is not retried.

Chaining check: Test for command reject. If not present, retry the operation 10 times.

Environmental data present: Issue a BUFFER UNLOAD command and retry the operation.

Track condition check: On CP I/O and Diagnose I/O, when a track condition check is received from a device for which CP does not provide alternate track software recovery, the condition is treated as a permanent error. CP does provide alternate track support for other devices; this support is described in the section "Alternate Track Recovery, ERP (DMKTRK)."

The error recovery routine keeps track of the number of retries in the IOBRCNT field of the IOBLOK. This count determines if a retry limit has been exceeded for a particular error. On initial entry from DMKIOS for an error condition, the count is zero. Each time a retry is attempted, the count is increased by one.

The ERP preserves the original error CSW and sense information by placing a pointer to the original IOERBLOK in the RDEVBLOK. Additional IOERBLOKs, which are received from DMKIOS on failing restart attempts, are discarded. The original IOERBLOK is thus preserved for recording purposes.

If after a specified number of retries, DMKDAS fails to correct the error, the operator may or may not be notified of the error. Control is returned to DMKIOS. DMKIOS is notified of the permanent error by posting the IOBLOK (IOBSTAT=IOBFATAL). The error is recorded via DMKIOS by DMKIOERR, if DMKDAS and DMKIOE determine that the error warrants recording.

If the error is corrected by a restart, the temporary or transient error is not recorded. Control is returned to DMKIOS with the error flag off.

Before returning control to DMKIOS on either a permanent error or a successful recovery, the ERP frees all auxiliary storage gotten for recovery CCWs, buffers, and IOERBLOKs, and updates the statistical counters for 2314 and 2319 devices.

The DMKIOS interface with the ERP uses the IOBSTAT and IOBFLAG fields of the IOBLOK to determine the action required when the ERP returns to DMKIOS.

When retry is to be attempted, the ERP turns on the restart bit of the IOBFLAG field. The ERP bit of the IOBFLAG field is also turned on to indicate to DMKIOS that the ERP wants control back when the task has finished. This enables the ERP to receive control even if the retry was successful and allows the freeing of all storage gotten for CCWs and temporary buffers. The IOBRCAW is set to the recovery CCW string address.

In handling an intervention-required situation, the ERP sends a message to the operator and then waits for the device end to arrive. This is accomplished by a return to DMKIOS with the ERP bit in the IOBFLAG field set on and the IOBSTRT bit in the IOBFLAG field set off. When the device end interruption arrives, the original channel program which was interrupted is then started.

The ERP flags of the IOERBLOK are also used to indicate when special recovery is being attempted. For example, a READ HOME ADDRESS command when a no record found error occurs.

The other two indicators are self-explanatory and are explained in Figure 22.

Field			
IOBFLAG IOBERP	IOBFLAG IOBRSTRT	IOBSTAT IOBFATAL	Action To Be performed by DMKIOS
1	0	0	Return control when solicited device end arrives
1	1	0	Restart using IOBRCAW
0	0	1	Permanent I/O error
0	0	0	Retry successful

Figure 22. Summary of IOB Indicators

If the error is uncorrectable or intervention is required, the ERP calls DMKMSW to notify operator. The specific message is identified in the MSGPARM field of the IOERBLOK.

ALTERNATE TRACK RECOVERY, ERP (DMKTRK)

The software alternate track recovery support described in the following paragraphs applies only to the 3340/3344 disk. For 3330 and 3350 disks no software support is needed since the hardware performs alternate track recovery. No support is needed for the 2305 drum since the CE is able to rewire the device to use spare tracks in place of defective tracks. For the 2314 and 2319 disks no true alternate track recovery is provided by CP. But track condition checks from any device type are reflected back to the virtual machine. Therefore, even though CP itself cannot use a 2314 or 2319 cylinder that contains a defective track, it it possible for a virtual machine to use such a cylinder if it provides its own error recovery. To facilitate this, the VM/370 version of the IBCDASDI program allows 2314 and 2319 minidisks to be formatted with an alternate track cylinder as the last cylinder of each minidisk rather than using the last cylinders of the real disk for this purpose.

Overview of 3340 Alternate Track Support

The 3340 alternate track support applies to CP I/O, to Diagnose I/O (thereby giving alternate track support to CMS), and to SIO executed in a virtual machine. For CP I/O and Diagnose I/O, the alternate track recovery support essentially consists of directing (seeking) an interrupted channel program to an alternate track and restarting it. Later, in some cases, the interrupted channel program is directed back to the original cylinder and restarted there. For SIO in a virtual machine, the operating system in the virtual machine provides its own error recovery when CP reflects a track condition check to the virtual machine.

On the 3340 disk, alternate tracks are assigned in the conventional alternate tracks cylinders at the high end of the real disk, not in the last cylinder of each minidisk. Therefore a virtual machine may need to seek outside of its minidisk extent. This occurs when an operating system in a virtual machine performs its own error recovery following a track condition check. So for SIO issued from a virtual machine, CP's

alternate track support must permit the virtual machine to escape from the confines of its minidisk to get to the alternate tracks assigned to the defective tracks of that minidisk. Yet at the same time CP must still prevent the virtual machine from accessing other tracks that it does not own.

Since alternate tracks are assigned only in the conventional alternate tracks cylinders at the high end of the real disk, CP does not apply minidisk cylinder relocation values to a virtual machine's channel commands that reference alternate tracks. Similarly, CP does not unrelocate alternate track CCHH addresses returned by read home address, by read record zero, in sense information, or for error recording.

Alternate Track Hardware Operation and Implications

The home address record (HA) on any track contains a flag byte with two bits that are involved in alternate track assignments. One bit, when set to one, indicates that the track is defective and that the track should have (and ordinarily does have) an alternate track assigned. The count field of record zero of a track with this bit set should point to (have the CCHH address of) the assigned alternate track. The second bit in the flag byte, when set to one, indicates that the track in which it appears is an assigned alternate track. The count field of record zero of an assigned alternate track should point back to (have the CCHH address of) the flagged defective track that it is assigned to.

Before using the pointer in record zero of a flagged track to get to the corresponding alternate, it is considered good form for an operating system to check the pointers both ways to see that each points to the other. CP performs two-way checks of the pointers for seeks to an alternate track initiated by Diagnose or by SIO in a virtual machine. For its own I/O, CP uses the forward record zero pointer without performing a two-way check. Performing a two-way check would decrease performance and should not be necessary since all of the record zero pointers were checked in both directions by the Format/Allocate program (DMKFMT) when the CP-owned disk was originally formatted.

Note: the DASD Dump/Restore (DDR) program also checks the record zero pointers both ways when a tape is restored to a disk.

Except for those channel commands that deal specifically with the home address and record zero, any attempt to search or read or write on a track that is flagged as defective results in a unit check with "track condition check" indicated in the sense data.

Operations on an assigned alternate track can also result in a unit check with "track condition check" indicated in the sense data. But in this case it occurs when an attempt is made to leave the assigned alternate track, not when the operation is reading or writing on the track. The situations where trying to leave the alternate track results in a track condition check are:

- Any multi-track operation
- A record overflow operation

The hardware does not generate a track condition check when a seek is used to leave the track. This applies to any kind of seek, including seek head.

When a channel program from a virtual machine SIO (or from a Diagnose) is allowed to access an alternate track, subsequent CCWs in the channel program must be prevented from accessing adjacent tracks in

the alternate track cylinder since these may belong to other virtual machines. A channel program may attempt a transition from one track to the next by any of the following:

- Seek
- Seek head
- Multi-track search or read
- Record overflow

The full seek causes no problem: since it specifies the cylinder as well as the track, it causes the channel program to leave the alternate track and to return to a cylinder within the minidisk extent. It is certain to go back to the minidisk because the seek address was verified when the virtual CCWs were translated to real.

The multi-track operations and record overflow operations also cause no problem, because, as explained above, these are caught by the hardware and result in a track condition check.

The seek head is dealt with as follows. When a seek to an alternate track is encountered in a virtual channel program by CP during the CCW translation process, CP converts all seek head commands (in the real, translated CCWs) to an invalid CCW opcode (X'FF'). Then when the translated channel program is executed, it is interrupted (with a command reject) at each seek head CCW so that the track to which the channel program is seeking can be checked to see that it really belongs to the virtual machine that requested the I/O. Note that this only happens to channel programs that seek out of the minidisk to an alternate track.

Module Function and Control Flow

DMKTRKVA - When DMKCCWTR finds a virtual machine seeking out of its minidisk extent to what should be an assigned alternate track, it has to do a check of the backward record zero pointer to verify that the alternate belongs to that minidisk. So DMKCCWTR calls DMKTRKVA, passing the CCHH address of the alternate as input, and DMKTRKVA performs CP I/O to read record zero of the alternate and then returns the pointer found in record zero to DMKCCWTR.

DMKTRKFP - This is called by both DMKUNT and DMKVIO. Its function is to handle command rejects in channel programs initiated by virtual machine SIO when the channel program was found (by DMKCCWTR) to be seeking to an alternate track outside the minidisk extent. The command rejects result because, for these channel programs, any seek head commands have been invalidated (opcode changed to X'FF') in order to trap seek heads that might switch to another minidisk's track in the alternate track cylinder.

Note: Even though DMKCCWTR may also find Diagnose I/O channel programs that seek directly to an alternate track and invalidate the seek head opcodes on these channel programs, the command rejects resulting from these channel programs are handled by DMKTRKIN, not by DMKTRKFP.

DMKTRKIN - This routine performs alternate track recovery for CP I/O and for Diagnose I/O both when the Diagnose channel program results in a track condition check and when a command reject results from a seek head whose opcode DMKCCWTR made invalid. The routine has nothing to do with alternate track recovery for SIO issued by a virtual machine. But it does share a few small subroutines with DMKTRKFP.

DMKTRKIN is called only by DMKDASER, which in turn is called only by DMKIOS. These three routines work closely together during alternate track error recovery and the control flow back and forth between these routines is controlled to a great degree by flags in the IOBLOK and the IOERBLOK.

The control blocks of major concern in this area are the RDEVBLOK, the IOBLOK, and the IOERBLOK. When an error occurs and DMKIOS makes the initial call to DMKDASER (at the time of the first error associated with this IOBLOK), an IOERBLOK containing sense data has already been created; the IOBIOER field of the IOBLOK points to it. When DMKDASER gets control, it notices that this is a first call and it moves the pointer out of IOBIOER into RDEVIOER so that this first IOERBLOK, associated with the original error, can be kept over a period of time during which attempts may be made to retry the I/O operation. During these retries, further errors may cause new IOERBLOKs, pointed to by IOBIOER, to be sent back from DMKIOS. Generally speaking, RDEVIOER continues to point to the original IOERBLOK and new IOERBLOKs are created and sent back from DMKIOS after each retry that ends with an error. Generally, the new IOERBLOK from the failed retry is discarded before the next retry. But occasionally a new IOERBLOK is used by DMKDASER or DMKTRKIN to replace the original IOERBLOK, so it is pointed to by RDEVIOER and the first original IOERBLOK is discarded before the next retry. This happens when the new error is deemed to be more severe than the original (DMKDASER gives priority to channel checks) or when the original error gets corrected by a retry, but then the channel program fails on a later CCW (DMKTRKIN does this).

Control flow back and forth between DMKIOS and DMKDASER is controlled by the setting of the flags IOBERP, IOBRSTRT, and IOBFATAL, and has been described earlier in the section "DASD Error Recovery, ERP (DMKDAS)."

The control flow back and forth between DMKDASER and DMKTRKIN is controlled by the flags IOERRDR0 and IOERALTR and by a return code that DMKTRKIN passes back in register 1. Whenever either of the two flags is set, they cause DMKDASER to call DMKTRKIN whenever DMKDASER gets control (which in this case happens after a retry), even though there is no track condition check indicated in the new IOERBLOK. The IOERRDR0 flag indicates to DMKTRKIN that the retry being returned from was used to execute a channel program to read record zero. The IOERALTR flag indicates to DMKTRKIN that the retry being returned from is a restart of a user channel program (not strictly error recovery CCWs) that had a track condition check earlier. This means that invalidated seek head opcodes can be expected.

Details of Alternate Track Recovery for CP I/O and Diagnose I/O

Once a CP I/O or Diagnose I/O channel program has to be restarted because of a track condition check, the error recovery procedure invalidates (for Diagnose I/O only) all seek head opcodes in the channel program and sets the IOERALTR flag (indicating that alternate track error recovery is in progress) before proceeding. The IOERALTR flag remains set whenever any portion of the users channel program is being retried, until the channel program either ends successfully or ends with a permanent error.

Note: The flag does not remain set continuously; there are breaks while the error recovery procedure takes time out to use its own channel program to read record zero (the channel program is passed back to IOS as a "retry"). At these times the IOERRDR0 flag is set instead of the IOERALTR flag.

During the further execution of a Diagnose Channel program, invalidated seek head opcodes may be encountered once the IOERALTR flag is turned on. CP channel programs do not use seek head. The number of these opcodes encountered may be several, or none at all, depending on the user's channel program. Also, these invalidated seek heads may be trying to seek off of an assigned alternate track (usually to the next logical track) or they may have no involvement with flagged tracks at all, again depending on the nature of the user's channel program. Whenever the channel program is stopped by an invalidated seek head, a determination is made of whether or not it is trying to get off of an alternate track. This determination is made by looking at the current cylinder number (available in sense data taken at the time of the command reject) and seeing whether or not it falls within the alternate track cylinder area at the high end of the disk. If the seek head was not trying to get off of an alternate track, there is no problem and the subject channel program is restarted with a seek to the current cylinder and to the track specified by the invalidated seek head. If the seek head was trying to get off of an alternate track, record zero of the alternate track is read first to get the cylinder number of the defective track. Then the subject channel program is restarted with a seek to the cylinder of the defective track, but to the track specified by the invalidated seek head.

TAPE ERROR RECOVERY, ERP (DMKTAP)

Error recovery is attempted for user-initiated tape I/O operations to CP-supported devices that use the DIAGNOSE interface. The primary control blocks used for error recovery are the RDEVBLOK, the IOBLOK, and the IOERBLOK. In addition, auxiliary storage is used for recovery channel programs (repositioning and erase).

The interruption handler, DMKIOS, performs a SENSE operation when a unit check occurs. Tape errors are then passed to DMKTAP. The sense information associated with a unit check is contained in the IOERBLOK. If a channel check is encountered, the channel check interruption handler determines if retry is possible and passes control to the ERP through the I/O interruption handler.

When an error is encountered and ERP receives control, DMKTAP determines if this is the first entry into the ERP for this task. The IOBRCNT (IOB error count) field of the IOB is zero on initial entry. On this first entry, the pointer to the IOERBLOK is placed in the RDEVIOER field of the RDEVBLOK. This preserves the original error CSW and sense information for recording. Thereafter, IOERBLOKS are discarded before a retry is attempted or a permanent error is passed to IOS.

The ERP looks for two other specific conditions. If the error count field is not zero, entry must be due to a recovery attempt. Thus, it may be a solicited device end to correct an intervention-required condition or a retry attempt for either tape repositioning or channel program re-execution.

The ERP keeps track of the number of retries in the IOBRCNT field of the IOBLOK to determine if a retry limit has been exceeded for a particular error. If the specified number of retries fails to correct the error, the error is recorded and DMKIOS is notified of the permanent error by turning on a status flag in the IOBLOK (IOBSTAT=IOBFATAL).

If the error is corrected by DMKTAP, the temporary error is not recorded and control is returned to DMKIOS with error flags all off. When repositioning is required in order to attempt recovery, additional ERP flags are contained in the IOERBLOK to indicate paths for specific errors (that is, data check on write must reposition, erase, and then reissue original channel program).

All error recovery is started the same except for intervention-required errors. The IOBFLAG is turned on to indicate RESTART (IOBFLAG=IOBRSTRT), and the IOBRCAW (IOBLOK Restart CAW) is filled with the restart channel address word. In addition, an IOBFLAG flag is turned on to indicate that the ERP is in control so that control can be returned to ERP during all tape error recovery (IOBFLAG=IOBERP). In the case of an intervention required error, the ERP sends a message to the operator, and then returns to DMKIOS with indications that tell DMKIOS the ERP is waiting for a device end on this device. This is done by clearing the restart flag and returning to DMKIOS with only the IOBERP flag on.

When ERP has determined a permanent error situation or successfully recovered from an error, all auxiliary storage obtained for recovery CCWs, buffers, and IOERBLOKs is freed before a return is made to DMKIOS (see Figure 22 for a summary of the IOB indicators), also, the statistical counters for 2400, 3410, and 3420 devices are updated.

If the error is uncorrectable or operator intervention is necessary, ERP calls the message writer to write the specific message.

3270 REMOTE SUPPORT ERROR RECOVERY

Recovery from errors associated with binary synchronous lines, and the related channel and transmission control unit hardware is processed by DMKBSC. Recovery from errors associated with data and control processing by the remote station (the device) as defined by remote status and sense byte definition (see IBM 3270 Information Display Component Description,) is processed by DMKRGF. Control blocks associated with these errors are the CONTASK, the RDEVBLOK, the BSCBLOK, the NICBLOK, the IOBLOK, and the IOERBLOK.

The interruption handler, DMKIOS, performs a SENSE operation upon detection of a unit check condition (IOERBLOK). The related sense data is analyzed as it relates to the previous operation (CONTASK or BSCBLOK, whichever is applicable). If a channel check is encountered by the channel check interruption handler, the channel check interruption (DMKBSC) procedures determine if recovery can be attempted. If it cannot be retried, that operation is aborted and an appropriate message is sent to the system operator.

Depending upon the error encountered, ERP receives control and either DMKBSC or DMKGRA and DMKGRB determines if this is the first entry into the ERP for this task. The IOBRCNT (IOB error count) field of the IOB is zero on initial entry. On this first entry, the pointer to the IOERBLOK is placed in the RDEVIOER field of the RDEVBLOK. This preserves the original error CSW and sense information for recording. Thereafter, IOERBLOKs are discarded before a retry is attempted or a permanent error is passed to IOS.

The ERP looks for two other specific conditions. If the error count field is not zero, entry must be due to a recovery attempt. Thus, it may be a solicited device end to correct an intervention-required condition or a retry of channel program execution.

The ERP keeps track of the number of retries in the IOBRCNT field of the IOBLOK to determine if a retry limit has been exceeded for a particular error. If the specified number of retries fails to correct the error, the error is recorded and DMKIOS is notified of the permanent error by turning on a status flag in the IOBLOK (IOBSTAT=IOBFATAL).

If the error is corrected, the temporary error is not recorded and control is returned to DMKIOS with all error flags off.

When ERP has determined a permanent error situation or successfully recovered from an error, all auxiliary storage obtained for recovery CCWs, buffers, and IOERBLOKs is freed before a return is made to DMKIOS (see Figure 22 for a summary of the IOB indicators). Also, the statistical counters for 3270 are updated.

The Attached Processor Environment

Attached processor support is requested by specifying AP=YES on the SYSCOR macro. For a complete description of system generation considerations, see VM/370 Planning and System Generation Guide.

CP Initialization for the Attached Processor

IBM System/370 Principles of Operation, has a detailed discussion of prefixing that is necessary for understanding the initialization done for the attached processor.

PROCESSOR ADDRESSES

The CP initialization routine, DMKCPI, begins normal processing by storing the physical, main processor address -- usually X'00' -- in the IPUADDR field in the PSA at location absolute zero. (Prefixing has not yet been established.) The logical processor address is computed by doing a logical OR of the physical address with X'40' and is stored in the PSA in LPUADDR. The logical value is used by the CP LOCK manager to avoid using a zero value. The physical value is used for signaling between the two processors.

If AP=YES was coded on the SYSCOR macro, DMKCPI uses the SIGP function to see if the attached processor is available. If so, its physical and logical addresses are stored in the PSA in IPUADDRX and LPUADDRX, respectively. If the attached processor is not available, APUNONLN is set to 1. If the multi-processing option is installed, message DMKCPI959W is sent to the operator.

PSA SETUP

The top two 4K pages of storage are marked (in the CORTABLE) as being CP-owned and are used as the PSAs for the two processors. The addresses of these two pages are stored at PREFIXA and PREFIXB in the PSA at location absolute zero. DMKAPI copies the information from the PSA at location absolute zero to the new PSA locations. In the PSA designated for the attached processor, PREFIXA and PREFIXB are switched. Thus, on either processor PREFIXA always represents the current processor and PREFIXB the other processor. The values of IPUADDR, LPUADDR, IPUADDRX, and LPUADDRX are also switched so that IPUADDR and LPUADDR always contain the processor addresses of the current processor and IPUADDRX and LPUADDRX contain the other processor addresses.

LOCKING

To provide system integrity, VM/370 attached processor support is designed around one global system lock, a VMBLOK local lock, and several system local locks for specifically identified queues or modules.

Global System Lock

All of the control program runs under the global system lock except specifically identified paths. If the lock cannot be obtained, the function is deferred by storing the necessary information in the VMBLOK appendage and stacking that VMBLOK for later processing. That processor then takes a special unlocked path through the dispatcher to dispatch a new virtual machine. In some situations the processor cannot defer the requested function and spins on the lock until it becomes available.

To ensure system integrity along the special unlocked paths, various local locks have been defined. These locks are basically spin locks and are held for short periods of time.

VMBLOK Lock

Each VMBLOK contains one lock, called VMLOCK, which is used by routines that need to serialize certain virtual machine related resources. These resources include the following:

1. Any unlocked or unshared pages belonging to the virtual machine.
2. Any of the unshared translation or backing store tables defining the address space of the virtual machine.
3. Certain fields of the VMBLOK that are modified by routines that do not hold the system lock. Some of these fields are VMPSW, VMGPRS, and VMRSTAT.

The dispatcher obtains the VMBLOK lock before a virtual machine is dispatched and also before a CP request or an I/O request is unstacked. When a virtual machine is dispatched, the VMBLOK address of this virtual machine is saved in the processor's PSA in the field RUNUSER. Normally this virtual machine is also unlocked by the dispatcher when it is entered after an interrupt handler has finished processing. When RUNUSER is still locked, the PSA field LASTUSER is equal to RUNUSER. When RUNUSER is unlocked, LASTUSER is set to ASYSVM.

When a CP request or an I/O request is unstacked, the associated virtual machine is locked and the VMBLOK address is placed in register 11. When the dispatcher is entered after a CP request or an I/O request has been serviced, the virtual machine whose VMBLOK address is in register 11 is locked and will be unlocked by the dispatcher. This virtual machine may not be the same virtual machine that was locked when the CP request or the I/O request was unstacked.

A CP routine must lock another virtual machine for any of the following reasons:

1. The routine, or a routine it calls, accesses any unshared page of the virtual machine.

2. The routine, or a routine it calls, alters any field of the VMBLOK that is serialized only by the VMBLOK lock.

3. The routine, or a routine it calls, could be interrupted and an exit taken to the dispatcher.

The original VMBLOK lock must be released before gaining the new lock.

Figure 23 shows the modules that obtain the VMBLOK lock for a virtual machine other than the one requesting the service.

There are situations when a CP routine may access a virtual machine without locking it. If the CP routine, or any routine it calls, is only altering VMBLOK fields that are serialized by the system lock, locking the virtual machine is not necessary. For example, to process the SET PRIORITY command for a virtual machine, locking the virtual machine is not necessary since the altered VMBLOK field, VMUPRIOR, is serialized by the system lock. But to process the SET FAVORED command, locking the virtual machine is necessary since some of the VMBLOK fields altered, such as VMRSTAT, are only serialized by the VMBLOK lock.

DMKLOKFR _ Free Storage Lock

DMKLOKRL - Run List Lock

DMKLOKTR - Timer Request Queue Lock

DMKLOKDS - Dispatcher Queues Lock
- CPEXBLOK Queue Lock
  deferred execution blocks
  processor related blocks
- IOBLOK/TRQBLOK Queue Lock

These are system spin locks that are held for very short periods of time. The control program code that runs without the global system lock must manipulate these queues and these locks insure system integrity along the unlocked paths.

User-Defined Locks

If you have user-defined areas that are used by more than one virtual machine and you need to serialize their use, you will need to define your own locking conventions. You can use the LOCK macro to obtain and release a PRIVATE lock. VM/370 System Programmer's Guide has details on how to code the LOCK macro.

MACHINE CHECK HANDLER IN ATTACHED PROCESSOR

A machine check interrupt is initially handled without the global system lock. DMKMCH determines if the error requires system termination, virtual machine termination, or simply recording and continuation. If the system was in a wait state or a virtual machine was in control and the system is not to be terminated, the machine check handler requests the global system lock with the defer option. If the lock can be

Module	Action
DMKAPI	Locks the virtual machine that was last dispatched.
DMKBLDVM	Locks the virtual machine just built.
DMKCFO	Locks the virtual machine being set as favored.
DMKCNS	Locks the virtual machine associated with a real device block.
DMKCPS	Locks the virtual machine whose virtual device is being reset when a real device is halted.
DMKCPU	Locks each virtual machine in order to prepare the VMBLOK for uniprocessor mode.
DMKCPV	Locks the virtual machine whose storage is being locked or unlocked, or for whom accounting is being done.
DMKCSU	Locks the virtual machine receiving transferred spool files.
DMKDIA	Locks the virtual machine of the dialed system, the virtual machine of the line being dropped (DMKDIADR), or the virtual machine that owns the channel-to-channel adapter being coupled.
DMKGRF	Locks the virtual machine associated with a real device block.
DMKLOG	Locks the virtual machine being reconnected or the virtual machine being autologged.
DMKMID	Locks the virtual machines receiving messages at midnight.
DMKMSG	Locks the virtual machine receiving a message.
DMKMSW	Locks the system operator.
DMKNES	Locks each virtual machine active when the NETWORK SHUTDOWN command is processed.
DMKNLD	Locks the virtual machine associated with a real device block.
DMKPAG	Locks the virtual machine associated with a queued I/O request.
DMKPTR	Locks the virtual machine from which a page will be stolen.
DMKQCN	Locks the system operator.
DMKRGA	Locks the virtual machine associated with a NICBLOK.
DMKRGB	Locks the virtual machine associated with a CONTASK or a NICBLOK.
DMKRNH	Locks the virtual machine of the destination user for a console task or the virtual machine associated with a remote teleprocessing line.
DMKSPL	Locks the virtual machine receiving a transferred spool file or the virtual machine owning a spooled reader file.
DMKVCA	Locks the virtual machine of the coupled-to CTCA.
DMKVCH	Locks the virtual machine to which the channel is being attached, or the the virtual machine from which the channel is being detached.
DMKVDA	Locks the virtual machine involved in attaching or detaching a real device.
DMKVDD	Locks the virtual machine involved in detaching a real device.
DMKVMC	Locks the virtual machine to which the caller is communicating.

Figure 23. Modules that Obtain Additional VMBLOK Lock

obtained, normal DMKMCH processing continues. If the lock cannot be obtained, DMKMCH stacks a CPEXBLOK with CPMCHLK set and exits to DMKDSPRU. This CPEXBLOK causes processing to resume at DMKMCHSE with the global system lock held. Any machine checks that occur before the CPEXBLOK processing has completed are considered recursive machine checks and handled appropriately. If the control program was in control and the system is not to be terminated, the machine check handler saves status in the CPEXBLOK, set CPMCHLK and reloads MCOPSW. CPMCHLK is set to prevent the dispatcher from starting any new work on this processor until the machine check processing has completed.

DMKMCH passes control to DMKMCTPT if the system is running in attached processor mode and a decision has been made to terminate the system. In general, if a virtual machine was running when the machine check occurred, only that virtual machine is terminated.

DMKMCTPT determines if the system can continue and if the processor can continue. For the attached processor, if the machine check was not a clock error and the control program was not in control, the virtual machine running at the time of the error is terminated. If the machine check was a clock error on the main processor or the control program was in control on either processor, the other processor is signalled to stop and store status and a wait state PSW is loaded on the failing processor. An attempt is made to issue message 610W to the operator before the main processor is stopped. If the machine check was a clock error on the attached processor and the control program was not in control, the main processor is signalled via an external call to initiate automatic processor recovery with an indicator to continue processing.

The malfunction alert interrupt handler (DMKMCTMA) receives control from the external second level interrupt handler. If the malfunction alert came from the main processor, a 001 wait state PSW is loaded. If the malfunction alert came from the attached processor and a virtual machine was in control, an indication is set to terminate the virtual user and CPAPRPND is set for processor recovery. If the attached processor was in supervisor state, message 610W is sent to the operator and a 013 wait state PSW is loaded. If the attached processor was in a wait state, CPAPRPND is set for processor recovery.

The automatic processor recovery routine (DMKMCTPR) receives control from the external SLIH or the dispatcher. If the system is to continue processing, the vary processor offline routne (DMKCPUUP) is called. DMKCPUUP examines the chain of virtual machines for attached processor affinity and shared segment pointers. Any shared segment pointers for the attached processor are switched to point to the main processor shared segments. All the system control blocks and save areas necessary to run in attached processor mode are also freed. The time from the first timer request queue element is placed into the clock comparator for the main processor.

While preserving the maintained fields in the absolute zero area, the main processor's prefix storage area is copied to the absolute zero area and prefixing is stopped. The attached processor operational flag is turned off in the absolute zero area, and the prefix storage areas for the main and attached processors are freed. The pages and DASD slots held by the attached processor for shared segments are freed by DMKPGT and DMKPTR. A message (194I) is issued, and return is made to DMKMCTPR. For any virtual machines with affinity to the attached processor, DMKMCTPR resets the affinity for each, issues message 621I, and puts the machine in console function mode (if the virtual machine is not disconnected). If a virtual machine is to be terminated, the virtual machine is reset, messages 616I and 619I issued. Normal return causes the system to continue processing in uniprocessor mode.

The action that the machine check handler takes for a given situation is determined by the error itself, the operating environment of VM/370, and whether the system was performing a CP function or a virtual machine function -- or the system was not performing at all (a loaded wait state condition when the error occurred). Figure 24 clarifies the action the system takes for the given situations.

	VM/370 Processing			Virtual Machine Processing		
Error Condition	uniprocessor	attached processor		uniprocessor	attached processor	
		Main	Attached		Main	attached
Invalid machine check interrupt code	1	1	1	1	1	1
Invalid PSW data	1	1	1	1	3	3
Register, Program mask instruction address invalid	1	1	1	1	3	3
System damages	1	1	1	1	3	3
TOD or CPU Clock Errors	1	1	1	1	1	3,4
Multibit (solid) Storage error	1	1	1	3,2	3,2	3,2
Multibit (intermittent) storage error	1	1	1	3,2	3,2	3,2
Storage Protect Key (solid) failure	1	1	1	3	3	3
Storage Protect (intermittent) failure	2	2	2	2	2	2
Malfunction alert	5	1	1	5	1	3,4
Channel inoperative	1	1	1	1	1	1

Legend:
1 = load wait state PSW
2 = refresh for retry operation
3 = terminate the virtual machine
4 = automatic processor recovery
5 = Not applicable

Figure 24. Condition/Action Table for Uncorrectable Errors

Multiprocessor External Interrupts

For external interrupts that can occur in attached processor mode (time-of-day sync check, malfunction alert, external call, and emergency signal), DMKPSAEX gives control to DMKEXTSL. DMKEXTSL does the following for each kind of interrupt:

Malfunction alert

- Call DMKMCTMA, which will either load a disabled wait state on the appropriate processor or initiate automatic processor recovery, to allow the system to run in uniprocessor mode. If a user was running at the time of the malfunction alert he is terminated.

SHUTDOWN Emergency Signal

Issued to the attached processor prior to shutting the system down.

- Turn off APUOPER in each PSA to indicate that the attached processor is not operational.
- Load a 008 disabled wait PSW.
- Disable channel zero.
- Pass control to the dispatcher at DMKDSPRU.

QUIESCE Emergency Signal

- Give control to the dispatcher at DMKDSPRU, which will load a wait PSW that is enabled for external calls only.

SYNC Emergency Signal

Issued by DMKCLKMP when the clocks are no longer synchronized (low order synchronization).

- Give control to DMKCLKAP to synchronize the clock on the attached processor. If the set clock fails, the attached processor is terminated with a CLK003 abend.

CLKCHK Emergency Signal

- Give control to DMKCLKCC. If the clock on the attached processor is not synchronized with the main processor (high order synchronization) or is not set, then a flag is set to cause DMKCLKMP on the main processor to synchronize the clocks. The attached processor is then put in a wait state enabled for external interrupts. If the clock is not working, the attached processor is terminated with a CLK003 abend.

APR External Call

- Give control to DMKMCTPR to allow the system to run in uniprocessor mode.

RESUME External Call

Cancels a previous QUIESCE.

- Give control to the dispatcher at DMKDSPRU.

WAKEUP External Call

"wake-up" an idle processor.

- If the system was running a user, reload the external old PSW.
- If the system was not running a user, then try to obtain the SYSTEM lock.
- If the SYSTEM lock is obtained, give control to the dispatcher at DMKDSPCH.
- If the lock is not obtained, give control to the dispatcher at DMKDSPRU.

DISPATCH External Call

Inform the other processor of a processor related CPEXBLOK.

- Try to obtain the global system lock.
- If the system lock is obtained, go to the dispatcher at DMKDSPCH.
- If the lock is not obtained and the system was in a wait state, go to DMKDSPRU.
- If the lock was not obtained and the system was not in a wait state, reload the external old PSW.

Time-of-Day SYNC Check

- Call DMKCLKSC. DMKCLKSC signals the attached processor to quiesce. It then sends message DMKCLK970W to the operator and calls DMKCLKMP. DMKCLKMP issues a SYNC emergency signal to synchronize the clocks. DMKCLKSC issues a RESUME signal to allow the attached processor to continue.
- If the SYSTEM lock is held, go to the dispatcher at DMKDSPCH.
- If the SYSTEM lock is not held, go to the dispatcher at DMKDSPRU.

I/O Subsystem

The I/O subsystem of VM/370 runs under the global system lock on either the main processor (processor with I/O capability) or the attached processor (processor without I/O capability). The I/O first level interrupt handler (DMKIOSIN) is the only exception; it runs unlocked during its normal error free processing. The main processor owns all real I/O blocks (RCHBLOK, RCUBLOK and RDEVBLOK). DMKIOSIN always runs on the main processor because only the main processor can receive I/O interrupts. All other routines that set any fields within the real I/O control blocks or that are dependent upon the status of a real I/O control block remaining static, must use the SWITCH macro to force their processing to the main processor. All routines within DMKIOS with the exception of DMKIOSRW, reissue the SWITCH macro whenever loss of control is a possibility, to ensure that processing remains on the main processor.

The SWITCH macro tests to see if it is running on the attached processor or the main processor. If it is running on the attached processor it issues an SVC 24 to transfer control to the main processor and to resume execution at the next sequential instruction.

If DMKIOS receives an unsolicited interrupt or an I/O error for scheduled I/O, a call is made to DMKFREE for either an IOBLOK, CPEXBLOK, or an IOERBLOK. DMKFREE will ensure that control is returned to the processor that initiated the request.

Shared Segment

The shared segment suufunction of VM/370 (DMKATS, DMKCFG, DMKCFH, DMKPGS, and DMKVMA) runs under the global system lock on either the main processor or the attached processor. All protected shared segments are duplicated in a system that is generated for attached processor mode and that is initialized on a machine with the multiprocessing feature. DMKCFG obtains sufficient storage to construct the duplicate page and swap tables in contiguous storage. The SHRTABLE SHRPAGE pointer points to the page and swap tables for the main processor, and the page and swap tables for the attached processor are at a fixed displacement from the page and swap tables for the main processor. DMKCFG initializes both sets of page and swap tables. Initially, the two swap tables point to the DASD locations specified in DMKSNT. However, as the pages are read into storage and then stolen, each shared page is allocated its own DASD slot and is pointed to by only one swap table entry.

The last user to purge a shared system causes both sets of page and swap tables to be released.

One shared page table is reserved for use by each processor. This includes both problem state and supervisor state execution on behalf of a virtual machine. To accomplish this, each time a virtual machine running a shared system is locked, a test is made to determine whether or not the virtual machine was last serviced on this processor. If it was last serviced on the other processor, all of its shared page table pointers in its segment tables are switched to this processor's shared pages.

DMKPTR is able to steal a shared page from a shared page table reserved for the processor it is running on without notifying the other processor. The virtual page could not appear in the look-aside buffer of the other processor.

The dispatcher releases the VMBLOK lock on LASTUSER following the check for pending interrupts (assuming no fast redispatch possible) unless the virtual machine was running one or more shared systems. In the latter case the VMBLOK lock is not released until the DMKVMA scan for a changed page is completed.

DMKVMA scans all protected shared segments that the virtual machine used. For every changed page that it finds, DMKVMA checks whether or not the system lock is held. If the system lock is held, the changed page is returned to CP free storage. If the system lock is not held, DMKVMA marks the page table entry as invalid, marks the swap table entry as in transit, and indicates that the core table entry is on the free and flush lists. The other virtual machines can continue to use the shared segments. The changed pages are replaced when the next reference to the changed page is made.

If the shared segment is violated, an error message (DMKVMA456) is sent to the violator, and he is placed in console function mode. The user may examine his PSW and registers to determine what caused the violation. The user enters the BEGIN command to resume execution at the point of interruption.

CP Method of Operation and Program Organization

This part contains the following information:

- CP Program Organization
- Use of the Annotated Flow Diagram
- Virtual I/O Operations and Interruption Processes

Use of the Annotated Flow Diagram

The following text sections, which describe each major CP function, are annotated flow diagrams. These diagrams, consisting of logic labels and commentary, describe the general flow and use of CP logic modules and their relationship to other modules while performing a specific function or task. The annotated flow diagrams do not contain references to error messages, abnormal termination conditions, or most control block field labels. This avoids complexity and makes the general logic of CP and its related tasks more understandable to the user. With "understandability" as the key, obtuse and complex logic that is used for obscure and seldom used functions is not described. Also the flow diagram does not indicate nor describe every entry point encountered in a function. Nor do the diagrams illustrate the innumerable times that commonly used modules are utilized. DMKFRE and DMKCVT, the obtaining and returning of free storage and the number base conversion modules are such examples. Annotated flow diagrams are arranged by function and subfunction. Titles for these functions and subfunctions also precede annotated flow text and labels. The text in the charts is prefixed by underscored and capitalized entry points and labels. Entry points are indicated by seven or eight characters; the first three characters are DMK. Labels are indicated by prefixing with a comma and the six-character module identification.

| The annotated flow diagrams in this section do not reflect VM/370 use
| of the MSS. If there is an MSS attached to the VM/370 system, consult
| Appendix B in this volume for flow diagrams of those functions that
| utilize the MSS (such as logging on a virtual machine that has a
| mimidisk defined on an MSS 3330V volume).

Note: Annotated flow diagrams are not to be construed as trace material. The dynamics of CP operations preclude the use of the annotated flow diagrams, as they are shown in this manual, as traces of CP functions.

VM/370 CP Interruption Processing

SVC INTERRUPTIONS - PROBLEM STATE

DMKSVCIN
Entry for SVC interruptions from problem or supervisor states. For problem mode and ADSTOP (SVC X '83'), the overlaid instruction is replaced.

DMKCFMBK
Console function mode is entered.

DMKSVCIN
For problem state SVC 76 (X'4C') check for valid parameter passing.

DMKVERD, DMKVERO
Determine the operating SCP used in the virtual machine by examining passed parameters in R0 and R1.

DMKSVC, SVCVER
For invalid parameter passing, error recording is not performed.

DMKIOEVR
The SVC is reflected to the user.

DMKIOFVR
On correct parameter reflection, record the error.

DMKSVC, REFSVCB
REFSVCB is called if TRACE SVC was in effect or if the virtual machine's page zero is not in real storage. Obtains the system lock before continuing. If the system lock is not immediately available, REFSVCB defers the interrupt and exits to DMKDSPRU.

DMKTRCSV
The DMKTRC module is called if TRACE SVC was invoked.

DMKPRGRF
If tracing is not active, flag user as being in instruction wait state and reflect the SVC back to the user.

DMKSVC
If the virtual machine's page zero is in real storage, generate and store an old SVC PSW. Fetch the new SVC PSW. If there is no PSW state change, store user's new PSW in RUNPSW, restore registers and dispatch via LPSW.

DMKSVC, REFSVCA
If there is a PSW state change, obtain the system lock before continuing. If the system lock is not immediately available, defer the interrupt and exit to DMKDSPRU.

DMKDSPB
Check the altered PSW.

SVC INTERRUPTIONS - SUPERVISOR STATE

DMKSVC, SVCDIE
Entry is for a system failure and is a SVC 0 or SVC 4 abend condition.

DMKDMPDK
Perform partial or full real storage dump.

DMKCKPT
Checkpoint the system.

DMKCPINT
Perform an automatic IPL if indicated.

DMKSVC, SVCLINK
Entry via SVC 8 provides linkage to a called routine in R15.

DMKPTRUL
If called routine is not resident, page it in and return control to the caller by loading the SAVERTN into the old PSW and then load the old PSW. The caller's addressability, SAVEAREA address and return address are maintained in a new SAVEAREA.

DMKSVC, SVCRET
Entry via SVC 12 return control from the called routine to the calling routine and restores addressability via R12 and R13.

DMKPTRUL
If a nonresident module, unlock page to return it to DASD.

DMKSVC, SVCRLSE
Entry via SVC 16 to release the current SAVEAREA used by SVC 8 and 12. Return to caller.

DMKSVC, SVCGET
Entry via SVC 20 to obtain a new SAVEAREA. Return to caller.

DMKSVC, SVCSWIT
Entry via SVC 24 to switch control to the main processor.

EXTERNAL AND CLOCK INTERRUPTION REFLECTION

DMKPSAEX
Entered via the interruption key on system console, adjust accounting to charge for supervisor overhead. If problem mode, attention interruption, update the virtual machine PSW from the external old PSW.

DMKPSA, EXTBUTTN
Exit to dispatcher, if there is no logged-on operator, or the operator is disconnected, or there is no active terminal. If the operator was logged on and the external interruption key was pressed, disconnect the operator's terminal.

DMKQCNCL
Clear all console requests.

DMKSCNRD
If the device is a terminal or graphic device, issue HIO to the real device.

DMKDSPCH
Exit to the dispatcher.

DMKPSA, EXTBUTTN
For 3704/3705, convert resource identifier for the NCP terminal for the indexable entry into the NICBLOK for the associated VMBLOK, then

DMKRNHND
Reset all BTUs.

DMKDSPCH
Exit to the dispatcher.

DMKPSA, EXTEXTD
Upon location X'80' timer interruption, indicate the user end of the time slice by storing flag in the VMBLOK's VMOSTAT.

DMKDSPCH, DMKDSPRU
If the system lock is held or is available, exit to the main entry of the dispatcher, DMKDSPCH. Otherwise, exit to DMKDSPRU.

DMKPSA, EXTTIMER
Upon processor timer interruption, VMTLEVEL in VMBLOK as a real processor timer interruption.

DMKTMRVT
Simulate the interruption.

DMKDSPCH, DMKDSPRU
If the system lock is held or is available, exit to the main entry of the dispatcher, DMKDSPCH. Otherwise, exit to DMKDSPRU.

DMKPSA, EXTCKC
Upon clock comparator interruption reflection

DMKSCHTQ
Use the printer to unchain the active TRQBLOK. Call DMKSTKIO.

DMKSTKIO
Stack the block.

DMKDSPCH, DMKDSPRU
If the system lock is held or is available, exit to the main entry of the dispatcher, DMKDSPCH. Otherwise, exit to DMKDSPRU.

MONITOR INTERRUPTION PROCESSING

DMKMON
The VM Monitor data collection component uses both sample and trace techniques. Selected system counters are sampled by routines entered periodically via TRQBLOK. Selected events are traced upon execution via monitor call instructions embedded at strategic points in the control program.

DMKMONTI
TRQBLOK gives control here every 60 seconds (unless specified otherwise with the MONITOR INTERVAL command), and a new TRQBLOK is immediately stacked via call to DMKSCHST, to request a return to the same entry point 60 seconds later. Control register 8 monitor mask is used to determine which of the three sampled data classes (PERFORM, USER, DASTAP) are enabled. Appropriate counters are sampled using special subroutines for each class and the data is stored in the monitor buffers. Upon completion, goes to dispatcher.

Class	Code	Data Items Collected
0	0	Software and hardware utilization, contention, and activity counters
	1	Corresponding items for APU
4	0	VMBLOK user resource consumption and status
6	0,1	DASD and tape activity counters

DMKENTTI

Entered via TRQBLOK every two seconds (unless specified otherwise with the MONITOR INTERVAL command). A new TRQBLOK is immediately stacked via a call to DMKSCHST to specify return of control to the same entry point two seconds later. This subroutine is a high frequency (relative to the PERFORM, USER, DASTAP sampler) I/O status sampler. All channels are tested for a busy condition with a TCH instruction. All control units and devices are tested for a busy condition by examining the appropriate CP control blocks. The data obtained is accumulated for later sampling by the DASTAP class of data collection in a class 6 (DASTAP) code 2 (I/O status) record. The subroutine DMKENT62 performs this collection after the standard class 6 (DASTAP) code 1 record has been collected by MONCOD61 in DMKMONTI.

DMKMONMI

Entered from DMKPRG after a monitor call in a class currently enabled (as defined in CR 8 mask) has been executed by CP in supervisor state. The monitor call instruction number and code number stored by the hardware in the PSA are used to index branch tables to reach the appropriate data collection routines. As necessary, the data is stored in the monitor I/O buffers before output. Upon completion, control returns to instruction after monitor call.

Class	Code	Activity Being Monitored
1	0	Begin console read
	1	Console output
	2	End console read
	3	Console sleep
2	2	User dropped from queue
	3	User added to queue
	4	User added to eligible list
5	0	Privileged instruction being simulated
7	0	SIO for DASD SEEK
8	0	Add queue, drop queue - more detailed resource consumption data

DMKMONPR

All data collection subroutines use a common buffer management subroutine to obtain sufficient space in the monitor buffers. When not enough space is available, a switch is made to the next buffer in the chain and the full buffer is scheduled for output via a CPEXBLOK. I/O is handled by DMKIOSQR if tape is in use, or by DMKMIAWO if a spool file is in use. If data collection gets ahead of buffer output and all the monitor buffers are filled, a temporary suspension occurs.

DMKMONIO

Handles normal and abnormal completion of buffer output to disk or tape. For normal completion, the buffer used for I/O is made available for further data collection; if the next buffer is already full, its output is immediately scheduled. If a suspension was in effect, data collection is immediately resumed using the freed buffer. (Note: Suspensions should be eliminated by increasing the buffer allocation, using the MONITOR command or the SYSMON macro.) Special tape conditions that can be handled include end of tape and permanent error.

DMKENTKC
Entered via CPEXBLOK at midnight if automatic monitoring to spool file is in effect and it is required to close out the current file and continue monitoring with a new file. DMKENT satisfies the nucleus residency requirements of CPEXBLOK entry point and acts as a stepping stone to DMKMIA. Goes to DMKDSP after successful call to DMKMIAKC.

DMKMIAKC
Sets up a request to invoke a MONITOR CLOSE command in DMKMCCCL.

DMKMCCCL
Executes MONITOR CLOSE command and calls DMKMIACC to complete processing.

DMKMIACC
Invoked by the MONITOR CLOSE command to close the spool file and chain the spool file block to the reader of the virtual machine where data reduction is to take place. Starts new spool file if appropriate.

DMKENTST
Entered via TRQBLOK due to previous determination by automatic monitoring facilities that a MONITOR START SPOOL command should be issued. This entry satisfies the need for CP nucleus residency and immediately calls the pageable DMKMIAIN.

DMKMIAIN
Builds a message buffer containing a MONITOR START SPOOL command and calls DMKMCCCL.

DMKMCCCL
Executes MONITOR START SPOOL command. DMKENTST gives control to DMKDSP after successful execution.

DMKENTET
Entered via TRQBLOK due to previous determination by automatic monitoring facilities that a MONITOR STOP command should be issued at this time. This entry satisfies the need for CP nucleus residency and immediately calls the pageable DMKMIAEN.

DMKMIAEN
Buils a message buffer containing a MONITOR STOP command and calls DMKMCCCL.

DMKMCCCL
Executes MONITOR STOP command. DMKENTET gives control to the dispatcher after successful execution.

DMKMIAST
Entered from DMKCPI when it is determined that automatic monitoring has been requested via the SYSMON macro in DMKSYS and that TRQBLOKs should be queued via calls to DMKSCHST to invoke a MONITOR START SPOOL command and a MONITOR STOP command at specified times in the future. If monitoring is required to start immediately because the start time has passed, a CPEXBLOK is built to give control to DMKENTSC, which invokes the DMKMIAIN mechanism described above.

All other DMKMCC, DMKMNI and DMKMIA entry points are used as a result of the processing of MONITOR commands or special conditions.

Three Class 0 monitor call codes have been reserved for special purposes. They are used without actually executing monitor calls, but as a result of MONITOR command processing. They are:

Class	Code	Function
0	97	Write header record after MONITOR START command
	98	Write trailer record after MONITOR STOP command
	99	Write suspension record when data collection resumes

DMKPRGIN
For a program interruption received while in supervisor mode (indication of CP module error) and INTRDR+1 does not indicate MONITOR CALL (X'40') exit to -
DMKPRG, CPERROR
Send abend message to the system operator.
DMKDMKPK
Dump storage and initiate loading (via IPL).
DMKPRGIN
For supervisor state and MONITOR CALL save registers in in DMKPRGPR.
DMKPRGMI
Do MONITOR CALL interruption processing (DMKMON).
DMKPRG, PRNSTAT
For paging exception X'11' and EC mode with translation on call DMKVATEX.
DMKVATEX
Process the exception.
DMKPRGIM
For paging exception, x '11' and EC mode with translation off, and enabled for I/O interrupts and PAGEX on call DMKVATPF.
DMKVATPF
Process the pseudo page fault.
DMKPRG, PAGEXCP
For all other page fault conditions go to DMKPTRAN.
DMKPRG, OBSLOCK
The system lock must be obtained before DMKPTRAN is called. If the system lock is not immediately available, defer the interrupt and exit to DMKDSPRU.
DMKPTRAN
Bring in the page from the auxiliary device.
DMKDSPCH
Exit to dispatcher.
DMKPRG, PRNSTAT
For segment exception X'10' with EC mode on and translation on call DMKVATSX.
DMKVATSX
Process the exception.
DMKPRG, PRGSIMI
For the segment exception, X'10' does not follow the above parameters; process it as an addressing exception.
DMKPRG, TRANSEX
Process X'12' translation exceptions.
DMKPRG, PRG01
For privileged or operational exception of a virtual machine in supervisor mode, examine ITRPR+1 if X'01' or '02' call DMKPRVLG.
DMKPRVLG
Process the exception.
DMKPRV, DMKPRGSM
For virtual machines in problem mode, store the users new program PSW in VMBLOK VMPSW.
DMKPSASV
When the program interrupt occurs and the users page 0 is not resident or the virutal machine is in EC mode, paging is performed.
DMKDSPB
Check the new PSW.
DMKPRVLG
Validate the privileged operation indicated in VMINST and perform the service.

Code	Operation
X'08'	SSK - Set storage key
X'09'	ISK - Insert storage key
X'44'	EX - Execute instruction
X'80'	SSM - Set system mask
X'82'	LPSW - Load PSW
X'9C'	SIO - Start I/O
X'9D'	TIO - Test I/O
X'9E'	HIO - Halt I/O
X'9F'	TCH - Test Channel
X'AC'	STNSM - Store, then AND system mask
X'AD'	STOSM - Store, then OR system mask
X'B1'	LRA - Load real address
X'B202'	STIDP - Store processor ID
X'B203'	STIDC - Store channel ID
X'B204'	SCK - Set TOD clock
X'B206'	SCKC - Set TOD clock comparator
X'B207'	STCKC - Store TOD clock comparator
X'B208'	SPT - Set CPU timer
X'B209'	STPT - Store CPU timer
X'B20A'	SPKA - Set PSW key from address
X'B20B'	IPK - Insert PSW key
X'B20D'	PTLB - Purge TLB
X'B6'	STCTL - Store control registers
X'B7'	LCTL - Load control registers
X'BA'	CS - Compare and swap
X'BB'	CDS - Compare double and swap

DMKPRV, LOCKET
The system lock must be obtained before other supervisor routines are called. If the system lock is not immediately available, defer the interrupt and exit to DMKDSPRU.

DMKHVCAL
On privileged operations of DIAGNOSE X'83' and the associated function code, perform the service.

DMKVSIEX
Execute privileged I/O operations of SIO, HIO, TIO and TCH.

DMKTMRTN
Perform privileged operations related to TOD clock, TOD clock comparator and the processor timer.

DMKPRGSM
Program interruption is reflected back to the user on invalid instruction operands, unsupported instruction operand codes and DIAGNOSE '83' function codes that are not a multiple of 4.

Virtual I/O Operations and Interruption Processes

CTCA OPERATIONS BETWEEN TWO VIRTUAL MACHINES

DMKVSIEX
Virtual I/O operation is reflected to DMKVCA, the channel adapter module, for processing.

DMKVCAST
For SIO, check if the CTCA is coupled. If not coupled, call DMKDIASM.

DMKDIASM
Simulate return status.

DMKVCA, VCRSTART
For a coupled CTCA, analyze operations resulting in X-side (read) and Y-side (write) of the data transfer operation.

DMKVCA, VCASIOB
Detected interruptions are presented to users via stacked IOBLOKs and DMKSTKIO.

DMKVCATS
CTCA TIO activity is determined by examining Y-side information to determine mode and activity.

DMKVCASH
CTCA HIO and HDV is processed by determining the conition code to present and whether the Y-side should be notified.

DMKVCARD
CTCA process results from RESET xxx or SYSTEM RESET commands. The CTCA status is reset but the CTCAs are not uncoupled.

DMKVCARS
Uncoupling CTCA is achieved in the VDEVBLOK (VDEVNRDY flag) idle CTCA plus an invoked DETACH xxx or user LOGOFF. Return to calling routine.

SCHEDULING I/O FOR CP AND THE VIRTUAL MACHINE

DMKIOSQR
Entered via SVC. Entry point indicate a CP I/O event as indicated in the IOBLOK. For start request, increment the SIO count in the RDEVBLOK and start the device if it is available. If not (device busy or already scheduled) queue the IOBLOK and return the operation to the caller.

DMKIOSQV
Entered via SVC. Entry point indicates virtual machine initiated I/O event. Preserve VMBLOK address in R11, turn off IOBCP bit in the IOBLOK, add 1 to SIO count in the VDEVBLOK (or RDEVBLOK). Process the SIO if there is any available path to the device. If not, queue the IOBLOK and return the operation to the caller.

STANDARD DASD I/O INITIATED VIA DIAGNOSE

DMKDGDDK
Perform simple disk I/O of a standard format. Entry is via DMKHVC code X'18'.

DMKSCNVU
Find device related to SIO cuu address.

DMKFREE
Allocate storage for IOBLOK and RCWTASK.

DMKGDDK
Build and check the CCW string.

DMKIOSQV
Execute I/O. On completion, post condition code (and error return code in R15, if detected).

DMKDSPCH
Exit to dispatcher.

GENERAL I/O OPERATION INITIATED VIA DIAGNOSE

DMKGIOEX
Perform general I/O operation. Entry is via DMKHVC code 20.
DMKSCNVU
Find device related to SIO cuu address.
DMKFREE
Allocate storage for the IOBLOK.
DMKCCWTR
Build the read CCW list.
DMKIOSQV
Queue the I/O request for execution.
DMKGIO, DIAGRTN
On interruption return, check status.
DMKUNTFR
If no problem encountered, free storage used for CCW string and IOBLOK.
DMKGIO, DIAGRTN
Reflect the condition code and return code to the user.
DMKDSPCH
Exit to dispatcher.
DMKUNTRN
On returned error condition, convert real CSW to virtual CSW and set in user's page 0.
DMKGIO, GIOEXT
Exit via SVC 12.

VIRTUAL MACHINE I/O INSTRUCTION SIMULATION AND INTERRUPTION REFLECTION

I/O Instruction Simulation

DMKVSIEX
Entry from DMKPRV to simulate I/O per VMBLOK's VMINST field.
DMKVSI, VIOSIO
On detected SIO, call -
DMKSCNVU
To locate VCHBLOK, VCUBLOK, and VDEVBLOK for the cuu called per SIO instruction.
DMKVSIEX
Determine device availability and set condition code accordingly.
DMKIOSQV
If the operation is warranted, schedule the operation.
DMKVSI, VIOTIO
For TIO, check device status, pending interrupts, and set appropriate condition codes.
DMKVSI, VIOHIO
For HIO, check for dedicated channel, CE, CU, or device busy condition, and subchannel busy and set appropriate condition codes.
DMKVSI, VIOTCH
Check for dedicated selector or busy channel and check for pending abnormal interruption and set appropriate condition code.

Interruption Reflection

DMKVIOIN
Entry from DMKDSP to process the reflected virtual interruption.
DMKSCNVU
Locate the VCHBLOK, VCUBLOK, and VDEVBLOK.
DMKVIOIN
Analyze blocks and reflect condition code to user. If condition code equals 1 (cc=1), save status from the real device (if real device) and DMKUNTFR.
DMKUNTFR
Translate and store CSW in user's page 0.
DMKVIO, VIOCC1
On TIO or HIO, free the device and set CC=1.
DMKFRET
Fret storage for the IOBLOK.
DMKDSPCH
Exit to dispatcher.

VIRTUAL CONSOLE SIMULATION

DMKVSIEX
Entry for virtual console activity comes from the SCP stored in the user's virtual machine. The program's generated CCWs and data are reflected to the attached terminal used by the virtual machine operator.
DMKVCNEX
Locate and move non-TIC CCWs from the users virtual storage to a VCONCTL block.
DMKVCN, GETCCW
Update CAW and CSW in respective control block.
DMKVCN, VCNRD
For read operation, build a read console buffer VCONBUF for the input to be read from the terminal.
DMKQCNRD
Queue a console read request.
DMKVCNEX
Set return address in VCONCTL VCNRDRET field.
DMKVSPVP
Spool console activity if SPOOL CONSOLE START specified.
DMKDSPCH
Exit to dispatcher. Wait for completion.
DMKVCN, VCNWR
Calculate and obtain free storage (VCONBUF) necessary for the write to console operation.
DMKVCN, VCNMDAT
Translate and bring in user's data page and move it into VCONBUF.
DMKQCNWT
Queue a console write request.
DMKDSPCH
Exit to dispatcher.
DMKVCN, VCNSNCN
ON a sense operation, set CE and DE in the virtual PSW. Reflect the PCI flag in the PSW if the PCI flag was set in the CCW. Set the IL flag if warranted. Move the sense data from the VDEVBLOK to user storage as designated by the CCW. Update VDEVBLOK's VDEVCSW to reflect status and count.

DMKVCN, VCNCC1
On completion of I/O operation, set appropriate status for command reject, not ready protection check, incorrect length, channel program check. Set appropriate CC and CSW in users page 0. Otherwise post pending interruption status in VMBLOK, VCHBLOK, VCUBLOK, and VDEVBLOK.

DMKVCN, FLAGTEST
If command chaining, process the next CCW.

DMKDSPCH
Exit to dispatcher.

LOCAL GRAPHIC I/O AND INTERRUPTION PROCESSING

DMKGRFEN
Entry for local graphic device enable and disable function (from DMKCPVEN and unstacked CPEXBLOK). Invoking CP ENABLE/DISABLE commands, start or terminate local 3270 display (and supported print devices) and certain system console activity.

DMKFREE
Performs enabling function. Gets storage for IOBLOK and TRQBLOK generation.

DMKGRF, LOGUSER
Form and write out the logo at the screen.

DMKGRF, ATTNINT
Unsolicited attention for RDEVBLOK (enabled).

DMKBLDVM
Build LOGON VMBLOK for logon process.

DMKCFMBK
Enter console function mode for terminal input.

DMKIOSQR
Schedule request to clear screen preparatory to logon.

DMKDSPCH
Exit to dispatcher to wait for interruption. Successful logon per the next interruption begins the operation of building the user's virtual machine.

DMKSCNRU
From the IOBLOK, locate the real device blocks related to the interruption. Analyze IOBLOK CSW and condition code and the I/O operation to determine read/write sequential action. For unit error, retry 10 times (if applicable). If recovery fails, log off. For ATTN interruptions, attempt to log on the new user if unsolicited ATTN occurs. Otherwise, set up for READ CCW string.

DMKFREE
Get storage for function and build CONTASK, IOBLOK, TRQBLOK.

DMKIOSQR
Issue the SIO.

DMKDSPCH
Wait for the response.

DMKGRFIN
Local 3270 display and certain system console interruption entry from dispatcher. On response of CE and DE, go to auxiliary processing routine address in TRQBLOK extension TRQBCRT and execute the processing routines:

Routine	Function
CONRETBF	Completion of a write CONTASK
RDMINT	Completion of a buffer read
GRFCFM	Execute console function
SETREJ	Set no accepted timer
SETMOR	Set more... timer delay
SETWNG	Set 10 second clear warning
RDEXIT	Clear buffers after PF keys
STRTREAD	Set read status
NOCTL	Process next CONTASK or go idle

DMKGRF, RDATA
Process read response of data plus ENTER key.
DMKCNSED
Edit and modify length count. Move data to caller's buffer.
DMKQCNWT
Schedule rewrite to screen (unless inhibited).
DMKIOSQR
Perform start I/O.
DMKDSPCH
Exit to dispatcher.
DMKGRFIC
Entry point to process CONTASKS queue for local 3270 and 3066 devices.
DMKFREE
Get storage for IOBLOK and TRQBLOK.
DMKGRF, BLDCCWS
Execute CONTASK, if appropriate. If not -
DMKDSPCH
Exit to dispatcher.
DMKGRF, RDMINT
For read return, determine function key action and write response (if appropriate) via KEYTEL.
DMKGRFTI
Entry point for processing timer interrupts.

LOCATE AND VALIDATE AN ISAM READ SEQUENCE

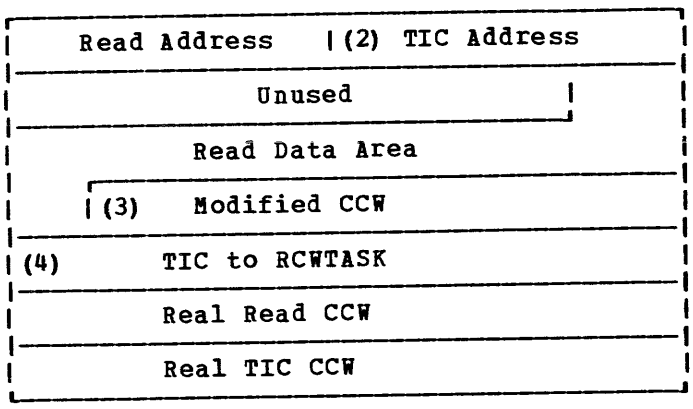
DMKISMTR
Entry from DMKCCW modules to locate and modify an ISAM CCW string. Using the IOBLOKs IOBCAW locate the RCWTASK. Check for the ISAM read CCW.
DMKISM, CHKRD
Check for the correct ISAM sequence as follows:

1. The last CCW in the RCWTASK is a TIC.
2. This RCWTASK points to the next RCWTASK with a minimum of 2 CCWs.
3. The first modified CCW is in real storage.
4. The last byte of the ISAM read overlays the operation code of the first CCW in the next RCWTASK.
5. The TIC in the RCWTASK is to the next RCWTASK's first CCW.
6. The date address of the first CCW in the next RCWTASK is the same address of the ISAM read+1 as it is in real storage.

DMKFREE
Storage obtained for seven double words save block.
DMKISM, CHKTSK2
Institute the ISAM read modification as follows:

1. Set the read to point to the save block data area.
2. Set the CP TIC to point to the modified CCW in the same block.
3. Set the modified CCW (seek head) in the save block to point to the save block data area.
4. Set the CP TIC in the save block to return to the RCWTASK following the modified (seek head) CCW.
5. Set the search CCW in the RCWTASK to point to the data area in the same block.

DOUBLEWORD SAVE BLOCK



DMKISM, CHKTSK2
Return to DMKCCW module via SVC 12.

SCHEDULING CP AND VIRTUAL MACHINE I/O OPERATIONS AND INTERRUPTION HANDLING

DMKIOSQR
Entry to process CP generated I/O. Flag the IOBLOK as a CP generated event. Initiate I/O if path to real device is free (available). If not, queue the IOBLOK and return to caller.

DMKIOSQV
Entry to process I/O for virtual machine I/O operations. Mark IOBLOK as not CP initiated. Save VMBLOK address. If path to the VDEVBLOK or the VDEVBLOK is busy queue the IOBLOK and return to caller.

DMKIOS, IOSTATDV
If available status, start the I/O and return to caller.

SIO Operations

DMKIOS, IOBSTART
If I/O request has not been reset, save the address of the active IOBLOK and set device busy. If the device is being reset, unflag scheduled device and scheduled control unit. Stack the IOBLOK and restart the device.

DMKIOS, IOSSIO
Set the subchannel path busy and chain the active IOBLOK from the RDEVBLOK.

DMKIOS, IOSSIO
Locate caller's CAW and issue the SIO. Check SIO completion. Returned condition code sets sequel action.
cc = 0 indicates successful start
cc = 1 CCW stored, initiate sense operation
cc = 2 Busy condition, retry or requeue IOBLOK
cc = 3 Fatal error (not operational), stack the IOBLOK and return to caller

HIO Operations

DMKIOSHA
Entry point for halting a device. If device is not active, return to caller. If IOBLOK active, reset the IOBLOK to halt the device and mark the device reset in RDEVBLOK.

DMKIOS, IOS1OKI
If the channel path is busy with a burst mode operation, stack the IOBLOK to halt the operation when the channel path becomes available. Return to caller.

Interruption Processing

DMKIOSIN
Entry from I/O new PSW. Check old PSW. If problem mode, save processor status in the VMBLOK.

DMKSCNRN
Locate RCHBLOK, RCUBLOK, and RDEVBLOKs for interruption unit.

DMKVIOIN
Process dedicated channel interruption condition. If control unit end or channel available interruption occurs, restart the operation, if interruption does not occur stack it.

DMKIOSIN
If the IOBLOK is not active on RDEVBLOK interruption, call DMKIOS.

DMKIOS, IOSENSE
Schedule sense operation, then go to dispatcher.

DMKIOS, IOSRSTRT
For PCI or CE interruptions, copy and stack the IOBLOK.

DMKCNSIN
Process PCI or CE interruptions, if related to local graphic device or nondedicated TP line.

DMKIOS, DOSENSE
For split seek complete interrupt, rechain the seek and reschedule operations.

DMKSTKIO
Stack IOBLOK and restart any units freed by the interruptions.

DMKDSPCH, DMKDSPA
If the system lock is held or is available, exit to the main entry of the dispatcher, DMKDSPCH. Otherwise, exit to DMKDSPA to try to redispatch RUNUSER.

TERMINAL CONSOLE I/O CONTROL, START/STOP, 3210, 3215, AND OTHERS

Enabling/Disabling

DMKCNSEN
Per unstacked CPEXBLOK, on enable or disable function, check current status of the current real device and set flag in RDEVFLAG. Build CONTASK and IOBLOK.

DMKIOSQR
Issue SIO for enabling or disabling function and check return.

DMKDSPCH
Exit to dispatcher.

Process CONTASK data

DMKCNSIC
Entry from DMKQCN module. Build I/O CCW string as defined by the console device type. Also select the proper line code to interface with the device. Place in CONTASK. For output CONTASK determine the correct translation table applicable to terminal communications (DMKTBL). To append proper control character to the data stream for the particular device type, refer to the following labels:

- DMKCNS, INCWTTY
  Teletypewriters
- DMKCNS, INC2741
  2741, 3767
- DMKCNS, INC1050
  1050, 1051
- DMKCNS, INC3210
  3210, 3215

DMKCNS, INCFINS
Attempt to start I/O by halting the current operation, if the operation is a "prepare" CCW or the input is a read and the forthcoming output is a priority write CONTASK.

DMKFREE
Get storage to build IOBLOK, if needed.

DMKCNSIN
Set return address in IOBIRA.

DMKIOSQR
Start I/O. If busy condition encountered build CPEXBLOK and queue for later execution.

DMKDSPCH
Exit to dispatcher.

Start/Stop Terminal Interruption Process

DMKCNSIN, CMBREAK
For an active input task halted, RDEVFLAG=RDEVHIO to process priority output task.

DMKFREE
Build CONTASK for reverse break CCWs.

DMKCNS, CNSBREAK
Move the input CONTASK following the last priority write output CONTASK on the chain.

DMKCNS, CNSIOUC
For unit check with intervention required, assume an attention interruption and build a "prepare" CCW for the 2741.

DMKCNS, CNSLOGF
For unit check and timeout condition - logoff the virtual machine and re-enable the line.

DMKCNS, CNSRTRY
For data check and other conditions, retry the previous operation.

DMKCQNET
Process completed output CONTASK.

DMKCNSIN
Interpret interruption status and CCW residual count for input CONTASK completion.

DMKCNS, CNINCT
Validate input data and control characters and translate to EBCDIC from line code.

DMKTRMID
Attempt to identify, if applicable, the line code identification; PTTC/EBCD or correspondence.

DMKCNSED
Perform line editing of the input buffer.

DMKCNS, CNSRT41
Prepare and issue control CCWs to request status information from the terminal.

Processing the Control CONTASK Interruption

DMKCNSIN, CNSCTAK
For control task interruption return, examine the interruption status according to control task function:

- DMKCNS, CNSTAK
  Reset control task.
- DMKCNS, CNSCTID
  Device identification.
- DMKCNS, CNSCTPR
  Attention signal.

DMKCNS, CNSCTPR
Write "VM/370 Online" interpretation of response determines retry, or build new CONTASK and execute or stack or process next CONTASK.

DMKQCNET
Process completed CONTASK requests. If no tasks remain for the terminal, set IOBLOK's IOBIRA to DMKCNSIN and link the IOBLOK to the user.

DMKDSPCH
Exit to dispatcher.

CONSOLE SCHEDULING

DMKQCNRD
SVC entry to build CONTASK for input data. Set the input buffer to zeros.

DMKFREE
Get storage to build CONTASK.

DMKQCN, ENQUEUE
Stack CONTASK on RDEVBLOK, if RDEVCON was zero. If not, exit to the appropriate interrupt handler per RDEVTYPC and RDEVTYPE or -

DMKSPCH
Exit to dispatcher.

DMKQCNWT
SVC entry to build CONTASK for output data. Strip trailing blanks from output message, modify byte count and determine real device destination.

DMKFREE
Get storage to build output CONTASK.

DMKQCN, WRDSCK
Update CONTASK CCW message byte count for the message text, terminal and line control information and (if appropriate) time stamp.

DMKCVTDT
If time stamp required, get the value for CONDATA area.

DMKVSPVP
Spool console message, if VDEVFLAG=VDEVCSPL.

DMKQCN, CRSCAN1
If message data contains carriage returns, X'15', create a separate CONTASK for each line.

DMKQCN, WAKEUPR
On first CONTASK or priority CONTASK, enqueue on chain from RDEVBLOK in appropriate location, then call related interrupt handler.

DMKQCN, WAKEMUP
If NORET or DEFRET specified, build and stack CPEXBLOK to alert the interruption handler and return via EXIT SVC otherwise go to specified interruption handler.

DMKQCNTO
Entry via SVC to disconnect and logoff a virtual machine as a result of transmission line failures. Place the virtual machine in a wait state, VMRSTAT=VMCFWAIT.

DMKSCHDL
Alter virtual machine to unrunnable state.

DMKFREE
Get storage for message for the system operator.

DMKSCNRN, DMKSCNRD, DMKCVTBH, DMKSYSNM
Fill in message variables.

DMKSCNR, DMKSCNRD, DMKCVTBH, DMKSYSNM
Fill in message variables.

DMKQCNWT
Send the user disconnect message to the operator.

DMKQCN, DSCGTRQ
Build TRQBLOK, if needed, for 15 minute delay, schedule it, and exit via SVC.

DMKQCN, DSCTLOG
After time elapse, TRQBLOK is unstacked and VMOSTAT is set to VMKILL for inevitable DMKUSOFF logoff operation.

DMKDSPCH
Exit to dispatcher.

3704/3705 INTERRUPTION HANDLER

DMKRNHIC
Entry via DMKQCN or via CPEXBLOK for 3704/3705 resource initialization. Locate the NICBLOK and check resource avaiability.

DMKRNH, LINEBRK
For resource unavailable, set RC=12 in CONTASK save area and return task via DMKQCNET.

DMKRNH, TAGTASK
For resource available, set CONTASK values per input and output task requirements.

DMKRNH, TASKENQ
Move CONTASK from RDEVBLOK chain to NICBLOK chain.

DMKRNH, RNSTART
On 3704/3705 available condition, search NICLIST and build an IOBLOK if required.

DMKRNHIC, RNEXLST
Search the NICBLOKs for CONTASKs to be sent to 3704/3705, build and chain for output.

DMKRNH, RNCHAIN
Perform necessary function for each resource.

DMKIOSQR
Start output I/O operations.

DMKRNH, RNICHN1
Return via R7.

DMKRNHND
Entry via SVC to schedule resource control tasks.

DMKRNH, RNHNDTK
Build control CONTASK and enqueue it for execution.

DMKRNH, STKCPEX
For NORET specified, build and stack a CPEXBLOK to perform SVC exit.

DMKRNH, RNDEXIT
Attempt to start output via GOTO DMKRNHIC.

DMKRNH, RNFDISC
Entry for 3704/3705 recovery.

DMKNLDR
Load the 3704/3705, if it was not previously loaded.

DMKFRE
Get storage to build CKPBLOK (telecommunications control block), if necessary.

DMKRNH, RNSBITS
Record active line and enabled terminal flag bits.

DMKQCNET
Clear CONTASK chains.

DMKQCNTO
Force disconnect to all active users.

DMKNLEMP
DUMP the 3704/3705.

DMKNLDR
Reload the named program.

DMKRNHND
On "IPL complete" signal, reenable resources.

DMKFRET
Release the CPEXBLOK.

DMKDSPCH
Exit to dispatcher.

DMKRNHIN
Entry via IOBLOK to perform input and output interruption processing.

DMKRNK, RNIOERR
For input process failure. Analyze the failure and if related to the 3704/3705 and not to a particular resource, either retry or dump and reload.

DMKRNH, READBUF
Interpret response codes for each BTU received and schedule necessary control operations.

DMKRNH, CMPREAD
Generate response to a read error.

DMKRNH, CMPWRITE
Generate response to a write error.

DMKRNH, CMPCONT
Generate response to a contact task error.

DMKRNH, COMDISC
Generate response to a disconnect task error.

DMKRNH, COMCNTL
Generate response to a control task error.

DMKRNH, UNSOLIT
Generate response to a unsolicited read.

DMKQCNET
Return completed CONTASKs.

DMKRNH, RNSTART
Attempt to restart the 3704/3705.

DMKDSPCH
Exit to the dispatcher.

DMKRNHIN
Entry via IOBLOK to perform input and output interruption processing.

DMKRNH, SCHREAD
On output, examine interrupt status per IOBLOK values and if ATTN, build and start a read CCW sequence.

DMKRNH, RNIOEUC
If unit check and fatal, dump and reload the 3704/3705.

DMKRNH, RNOREAD
If pending ATTN cleared via SIO -

DMKIOSQR
Reschedule write operations.

DMKRNH, RNSLOWDN
If unit exception, set RDEVSLOW and reschedule rejected CONTASKs.

DMKQCNET
Return only CONTASKs without CONRESP or CONSPLT set. Retain others until final response is received.

DMKRNH, RNSTART
Attempt to restart the 3704/3705.
DMKDSPCH
Exit to dispatcher.

HANDLING REMOTE 3270 WITH BINARY SYNCHRONOUS LINES

Remote Diplay Station and Binary Synchronous Line Enabling/Disabling

DMKRGBEN
Entered when the NETWORK ENABLE/DISABLE command is issued.
DMKFREE
Get storage for the necessary CONTASK, IOBLOK, and if applicable, BSCBLOK.
DMKRGB, LINESUP
Set up required CCWs and control data in the CONTASK for tasks. These tasks include: enabling the binary synchronous line, enabling a device, LOGO messages, screen formatting, and disable line or device (logoff).
DMKFREE
For logon function build logon VMBLOK.
DMKIOSQR
Start line I/O or device I/O, for not busy condition.
DMKRGB, RGFTASK
For busy condition, build CPEXBLOK and exit to caller.

Request Handler for 3270 I/O Events

DMKRGBIC
Entry from DMKDSP. On a not available line condition, exit to dispatch. For available line, process the associated CONTASKs by queueing the related resource from the NICBLOK.
DMKIOS, RGSTART
Process POLL SIO on a no CONTASK queued condition.
DMKIOSQR
Process selection SIO on available resources and not in control mode per NICBLOK conditions and the CONTASK CONSTAT field.
DMKDSPCH
Exit to dispatcher.

Secondary Interruption Processor for 3270

DMKRGAIN
Entry from DMKIOS, examine line interruption condition. Discard any of the following and go to the dispatcher: nonbinary synchronous line, copied IOBLOK, unsolicited interruption, bisync line flagged not-in-use, nonterminal class device.
DMKRGA, FATALER
For IOBFATAL condition or any nonzero condition code, free all related CONTASK, IOBLOK, IOERBLOK, and BSCBLOK.
DMKRGA, DISASTA
Log off all affected users on that line.
DMKMSWR
Send message to the system operator.

DMKDSPCH
Exit to dispatcher.

DMKRGAIN
If line or terminal response did not fall in the previous category, process via TP code branch. The code in the fifth byte of the ending CCW or IOBCSW-8.

TP Code	Function
TP00	Error Handling CCW
TP01	Enable/disable function
TP02	Write EOT (sequence prior to polling and addressing)
TP03	Write polling or addressing characters
TP04	Handle station's status and sense message
TP05	Read response to addressing
TP06	Write response to text
TP07	NO-OP following POLL command
TP08	Unit exception condition (timeout)
TP09	All reset commands
TP10	Read/write text
TP11	Read response to text

DMKDSPCH
Exit to the dispatcher.

3270 Binary Synchronous Line Error Recovery

DMKBSCER
Entry via DMKIOS and SVC 8 to process errors related to the binary synchronous line unit check and channel error conditions. On first error pass, move the IOERBLOK pointer from the IOBLOK to the RDEVBLOK, reset retry and fatal flags, set the ERP flag and call DMKFREE.

DMKFREE
Get free storage for a work area for retry CCWs.

DMKBSC, NOTFIRST
On a not first error condition, test for unrecoverable error condition. Unrecoverable errors include:
program check, protection check, chaining check, equipment check, interface control check and channel control checks. If one of these, notify the system operator. Reset flags, initiate error recording and

DMKFREE
Free IOERBLOK.

DMKIOSQR
Go back to scheduler.

DMKRGA
Analyze TP code, sense data CSW residual count and retry count to determine retry or IOBFATAL flag setting.

Process a Page Request

DMKPTRAN
Enter via the TRANS MACRO per paging request as determined by DAT created program interrupt (page or segment exception).

DMKPTR RESTART
Return to caller, if virtual address in R1 is beyond range of user's directory specified storage size.

DMKPTR, ADDROK
Check page residency via LRA (LOAD REAL ADDRESS) operation.

DMKPTR, TESTLOCK
For resident page, lock page in storage (if appropriate).

DMKPTR, GETRADD
Set real address in R2, make PAGTABLE entry valid. Set cc=0 and exit to caller.

DMKPTR, INTRAN
For page not resident but in transit (SWPTABLE, SWPFLAG), place virtual machine in locate mode. Locate CPEXBLOK for the real page requested and chain another CPEXBLOK with a return address of TRANRETN, to the same chain.

DMKPTR, TRANRETN
After page is no longer in transit, restore registers and return to RESTART for processing.

DMKPTR, GETPAGE
Reclaims a page on FREELIST (CORETABLE).

DMKPTR, DOIO
For page that is not in storage, do setup to read in the page.

DMKPTR, CKDEFER
For DEFER option passed in R2, build CPEXBLOK to return to user after page is in storage.

DMKPTR, PAGIN
After the page is read into storage DMKPAGIO process, remove the user from the wait state and update the lock count (if required).

DMKPTR, GETRADD
Set real address in R2, make PAGTABLE entry valid. Set cc=0 and exit to caller.

Obtain, Return, Lock and Unlock a Page of Free Storage

DMKPTRFR
Per the caller's code in R2, obtain a page frame -

DMKPTR, GETFREE
Obtain page frame via CORTABLE reference then exit to caller.

DMKPTRFE
Entry via CPEXBLOK, check page availability via flush list (DMKPTRFL), if none available steal a user's page.

DMKPTR, SELECT
The SELECT routine is entered to replenish the FREELIST from the flush list or user's pages that have not been referenced.

DMKPTRFT
Process pages to be returned by chaining them to the FREELIST. On page returns DEFER page requests are processed first.

DMKPTRLK
In locking a page in Real Storage (address in R2), add 1 to lock count; if previously locked, and exit to caller. If not previously locked, unchain the CORTABLE entry from the user's page list and set the lock count to 1.

DMKPTRUL
To unlock a locked page, reduce lock count by 1 and exit. If the lock count is now equal to zero, place CORTABLE entry on user's page list prior to exiting from routine.

READING/WRITING A DASD PAGE TO/FROM VIRTUAL STORAGE

Virtual Storage and Management - Non-EC Mode

DMKRPAGT
Entered via SVC call to read DASD page into storage.
DMKPGTPR
Release DASD space that was previously occupied by this virtual storage page.
DMKRPA, RESIDENT
Remove resident page frames from the user list.
DMKPTRFT
Place these page frames on the free list.
DMKRPA, STORDASD
Update the SWPTABLE with disk address in R0.
DMKPTRAN
Bring the page into storage.
DMKRPA, EXIT
Put real storage address of the virtual page is passed back to the caller in R2.
DMKRPAT
Entered via SVC call to write out a page to DASD storage.
DMKPTRAN
Locate the page to be moved and lock it.
DMKRPAPT
Store all registers in CPEXBLOK and flag CPEXR0 as a write request.
DMKPAGIO
Write the page.
DMKRPA, IORETN
Decrement page wait count. If zero results, take user out of page wait.
DMKPTRUL
Unlock the page frame. Return to caller.

Virtual Storage Management - EC Mode

DMKVATAB
Entry via BALR when an EC mode virtual machine needs a shadow table generation and update or purge operation.
DMKVATMD
Get storage to create shadow table, Flag VMBLOK to show shadow table existance.
DMKVATBC
Free shadow page, segment and copy segment, when user leaves EC mode or alters CR 0.
DMKVATRN
Entry to perform third level to first level translations and third level translations to second level address translations. Use TRANS macro to access virtual segment and page tables to get the virtual page into real storage.
DMKVATLA
Using the TRANS macro to access the virtual segment and page tables, pass the resulting page and displacement to DMKPRVLG.
DMKVATPX
Invoked by DMKPRGIN when a paging exception is received for an EC mode virtual machine.
DMKVAT, SETUPEX
Perform set up operation and develop page table address.
DMKPTRAN
Get the page.

DMKVATPX
Update the shadow table.

DMKVATSX
Invoked by DMKPRGIN when a segment exception is received for an EC mode virtual machine.

DMKVAT, SETUPEX
Perform setup operation, then invalidate the shadow page table or if none exists, allocate a new shadow table and set it invalid.

DMKVATPF
Entered via DMKVATPG from DMKPRG to simulate pseudo page fault interrupts when a paging exception occurs with pseudo page fault interrupts enabled.

DMKPTRAN
Bring in the DASD page.

DMKPRGSM
Reflect program check X'14' to the user.

DMKVAT, PAGRES
When the page becomes resident in storage. Build the PGBLOK, set high order bit in the translation exception address field,

DMKDSPCH
Exit to dispatcher.

ALLOCATION AND DEALLOCATION OF DASD SPACE

DMKPGTPG
Entry to search and allocate a DASD page for paging/spooling.

DMKPGTSG
Search appropriate RECBLOK chain for available DASD page. If none found, locate next available cylinder and construct a new RECBLOK, calculate address of the allocated DASD page and place it in R1. Return to caller.

DMKPGTPR
Entry to deallocate DASD page used for paging and spooling. Via RDEVBLOK locate the RECBLOK and reset appropriate bit in the RECBLOKs RECMAP and adjust the member of DASD pages in use. If all the pages on the DASD cylinder have been deallocated, deallocate the cylinder. Exit to caller.

DMKPGTSR
Entry to release a group of DASD pages no longer needed for spool file use. Per R1, find RECBLOK and dummy RECBLOKs and reset the RECMAP bits as specified. Free related RECBLOKS, if complete deallocation occurs.

DMKPGTCG
Entry for allocation of enough DASD spool space to record a 3704/3705 dump. Scan RDEVBLOK and associated ALOCBLOK for enough contiguous available space to record the dump. When found, flag cylinder as allocated and build and chain the required RECBLOKs.

DMKPGTVG
DMKPGT contains an internal table, PAGETABL, in which the allocation of page frames for the CP paging VMBLOK is kept. The PAGETABL is scanned for a zero bit denoting the page frame is available. The page is marked allocated by setting the bit to one and the address of the page frame is returned to the caller in R1. If no page frames are available, a CPEXBLOK is built and queued to the deferred request chain.

DMKPGTVG
Entry to release a page of virtual storage. Check the chain of deferred requests. If there are none, reset the page bit in the PAGETBL to 0 and exit to the caller. Otherwise, give the page to the first requestor in the deferred chain and stack his CPEXBLOK for the dispatcher.

SHARED SEGMENT STORAGE MANAGEMENT

DMKATSCF
Entry via SVC from the command processor if an ADSTOP, TRACE, or STORE command is to alter a shared page. The virtual machine issuing the CP command will be unshared from the named system, that is, given a private copy.

DMKERMSG
The running virtual machine is informed of the share page violation.

DMKVMASH
Entered from DMPDSP or DMKPTR via BALR. The protected shared page tables are examined for hardware change bit being on. The resulting condition code is reflected to the caller.

DMKVMASW
Entered to switch the virtual machine from one set of page tables to the other.

TEMPORARY DISK STORAGE MANAGEMENT

DMKTDKGT
Entry to allocate temporary disk space (T-disk). With R0 equal to the number of cylinders required and R1 equal to the device type, locate RDEVBLOK and related ALOCBLOK's ALOCMAP. If no allocation space is to be found, return to caller with 0 in R8. If allocation is successful, flag ALOCMAP, with X'AA' as allocated and put first cylinder address in R1 and RDEVBLOK pointer in R8 and return to caller.

PAGING I/O SCHEDULER

DMKPAGIO
Entry to initiate Page I/O activity. Using preformatted IOBLOK from IOBSTACK, fill in the CCWs with DASD opcode and values derived from CPEXBLOK swap table and core table. Chain the CPEXBLOK on the in-transit queue.

DMKPAG, GETRDEV
Find the Paging RDEVBLOK.

DMKPAG, FINDIOB
Search IOBLOKs seeking the same cylinder address. If found, chain the channel programs together with TICs.

DMKDSPCH
Exit to the dispatcher.

DMKPAG, QUEUEDIO
If no IOBLOKs with some cylinder address are found -

DMKIOSQR
Start the I/O operation.

DMKDSPCH
Exit to the dispatcher to await interrupt.

DMKPAG, UNTRANS
Upon interrupt return, unchain the CPEXBLOK from the intransit queue.

DMKSTPCP
Stack all deferred requests for execution.

DMKPAG, UNSTACK3
Return IOBLOK to IOBSTACK or free it.

DMKPAG, OVERHEAD
Calculate paging load and store it, the TOD, and other values in PSA.

DMKDSPCH
Exit to dispatcher.

RELEASE VIRTUAL STORAGE PAGES

DMKPGSSS
Entry to release partial virtual storage. Per R1 (address of first page to be released) and R2 (address of last page to be released) set partial entry flag.

DMKPGSPO
Entry to check for shared segments and decrement usage count. Store registers and flag full entry condition. Examine VMSHRSYS for shared segments. If so, decrement use count. On zero use count unchain the SHRTABLE from the active list.

DMKPGS, CKCLEAR
On NOCEAR exit to caller. If not, store number of release pages in R8.

DMKPGS, PGOUT2
Locate page and swap tables for the segment to be released and index to the entry for the first page.

DMKPTRAN
Initiate paging, and when paging stops release the page frame.

DMKPGS, NEXTPAGE
8 value.

DMKDSPCH
Exit to caller.

DMKPGSPS
Entry to release storage containing a named system passed by the caller. If register one is nonzero, search the page tables looking for a header equal to the named system. If found, release the swap and page tables and build new ones, if the address range still lies within the user's virtual storage size. If register one is zero, release and rebuild swap and segment tables for all segments above the normal virtual storage size that do not have SHRTABLE entries.

FREE STORAGE MANAGEMENT

DMKFREE
Entry to obtain a block of storage, validate input doubleword request (R0).

DMKFRE, FREESUB
ON subpool size request, index into SUBTABLE. For correct size block found, remove block from chain and put the address of the block in R1. Return to caller.

DMKFRE, FREE02
For subpool size not found get next large subpool size. Remove block from chain, put address in R1 and return to caller.

DMKFRE TRYSPLIT
For subpool that cannot honor request, start search a 30 doubleword end for block requirement. When a block is found, split block (if necessary) and give caller address of his portion in R1 and chain the remainder to the appropriate subpool size. Return to caller.

DMKFRE, CLEARSAV
If no block can be found to honor user request, call -

DMKPTRFR
Fetch a page from the dynamic paging area. Chain it to the free storage chain. Processing then continues. See entry DMKFRE, FREESUB.

DMKFRERS
Entry to return all subpool blocks to the free storage chain per the SUBTABLE reference, as each subpool block is released, its address and length are placed in R1 and R2 respectively. Branch and link to FRET05 to return the block to the free storage chain (DMKFRELS). Repeat action through all subpools. Return to caller.

DMKFRET
Entry to restore block to subpool or free storage. Per R0 and R1 (number of doublewords to be released and and address of the first double word, respectively), the subpool sized block is returned to the appropriate subpool. Update the pointer in the SUBTABLE.

DMKFRE, FRET21
If subpool size block being returned is within the dynamic paging area, process as a block of more than 30 doublewords.

DMKFRE, FRET20
Blocks larger than 30 doublewords to be returned are merged into the free storage chain indicated by DMKFRELs.

DMKPTRFT
Restore page to dynamic page area; if a complete page is alloted, blocks belonging to the dynamic paging area can be built.

DMKFRE, FRET03
Return a block of storage to free storage chain by merging into the chain storage addresses in an ascending order of sequence. Return to caller.

CP INITIALIZATION AND TERMINATION PROCEDURES

Loading the Nucleus

DMKCKPT
Initial entry point to load the system after loading the first module, DMKCKP, from the system residence volume. Check CPID in PSA for startup method.

DMKSAVRS
For CPID equal to not warm or not CPCP, insert COLD and load the nucleus. Then branch to DMKCPINT, to perform CP initialization.

DMKCKP, NOTERM
ON CPID equal to WARM or CPCP, halt and drain all I/O devices and remember enabled terminals.

DMKCKP, NEXTCH
DMKRSPCV to validate warm start cylinder.

DMKCKP, CLOCKOK
Save accounting data, log message, SDFBLOKs, and enabled terminals and lines on checkpoint cylinders.

DMKCKP, CHK05
Save spool records allocation and spool hold queue blocks on checkpoint cylinder.

DMKCKP, SHUTSYS
If normal shutdown indicated, issue message to system operator and load disabled wait state code X'008'.

System Initialization

DMKCPINI
Entry point to perform system initialization.

DMKCPI, KEYLOOP
Determine real storage size, initialize CORTABLE, allocate free storage and initialize system paging tables.

DMKCPI, CPIHIP
Check via HIO for online and ready status of all DMKRIO generated devices.

DMKCPI, CPISTCAW
Read volume labels and match to RDEVBLOK, RDEVSER.

DMKCIP, DMPALLOC
Allocate dump file to system device.

DMKCPI, ALOCLP
Build allocation block for CP-owned devices.

DMKCPI, MICTEST
Test for virtual machine assist feature availability If available, build MICBLOK and link to VMMICRO.

DMKCPI, NPSWS
Locate an available primary or alternate system console (PSA values).

DMKCPI, NOTCHNG
Build user directory page list per DMKSYSUD.

DMKLOGOP
Log on the system operator.

DMKCPI, STARTSYS
Force nonnucleus modules to DASD page.

DMKIOEFL
Initialize error recording cylinders.

DMKNLDR
Auto load 3704/3705; if appropriate.

DMKAPIPR
Initialize PSAs for each processor. Called only if the attached processor is available.

DMKCLKCK
Synchronize the TOD clocks if necessary. Called only if the attached processor is available.

DMKCPVAE
Enable 270X lines, if appropriate.

DMKCPI, CPIDSP1
Log on the AUTOLOG user.

DMKPTRUL
Unlock CPI as initialization is complete.

DMKDSPCH
Await interrupts.

Warm Start

DMKWRMST
Entry from DMKCPI initialization. Check R2=01; if it is, go to DMKWRN, WARMCLR for cold start. Check warm start cylinder for 8 byte X'FF's identifier.

DMKWRM, ENABLERT
If enable records on, warm start cylinder, enable appropriate RDEVBLOKs.

DMKWRM, EN370S
If warm start record indicates, set flag for auto load of the named NCP program.

DMKWRM, ENR3270
Enable binary synchronous lines by clearing NICBLOK offline flag (if appropriate).

DMKWRM, ACNTRT
Build ACNTBLOK, load it with warm start cylinder data and chain it.
DMKWRM, WARMLOG
Build buffer and load it with the saved log message.
DMKWRM, WARMSPL
Build SPFBLOKs and fill with appropriate printer, punch, and reader spool data.
DMKWRM, WARHOLD
Build SHQBLOK and move hold queue record data to the new block and chain it to the hold queue chain.
DMKWRM, WARMCLR
Clear 8 bytes of record 1 on the warm start cylinder. Check CPID again.
DMKCKSWM
For CPID=CKPT or FORCE, reconstruct spool checkpoint records.
DMKCKSIN
For CPID=NOT CKPT or NOTFORCE, initialize the checkpoint cylinders.
DMKCKSPL
Files in the systems spool hold queue are added to the checkpoint cylinder.
DMKWRM, GETDISK
Read in the remainder of warm start data.

Normal Shutdown

DMKCPSSH
Entry point results from invoking CP SHUTDOWN command. Close active spool files for callers or operator console.
DMKCPS, DASDCH
Via RDEVBLOK, locate and record DASD statistical data.
DMKCPS, DASDCHI
Put CPCP into CPID to denote shutdown.
DMKDMPRS
Set up CAW, CCWs and load CP via IPL from system residence device.
DMKCKPT
Save spooling and accounting data.
DMKMONSH
Stop monitor tape activity.
DMKCPI SHUTSYS
Sense shutdown flag, issue DMKCPI961W, enter disabled wait state code X'006'.

Dump the System

DMKDMPDK
Entry occurs via ABEND000 condition or by pressing system console RESTART button. Save PSA values. Determine if dump is full or just CP portion.
DMKDMP, DMPMSG
Format and issue abend message to operator and transfer to DMKDMP and DMPDASD.
DMKDMP, DMPDASD
Write out a defined amount of storage or all storage to selected DASD.
DMKDMP, DSKEND
Place sending record number and the system file number in the dump file SFBLOK.
DMKDMP, RECSRCH
Chain dump file RECBLOKs to RDEVBLOK, and link dump file SFBLOK onto the system reader chain.

DMKDSP RESTART
Restart the system on warm start indication.

DMKDMP, DMPTAPE
Dump CP storage or all storage to the selected tape drive per specified tape parameters.

DMKDMD RESTART
Restart the system if warm start is indicated.

DMKDMP, DMPPRT
Dump CP storage or all storage to the selected printer.

DMKDMS RESTART
Restart the system if warm start is indicated.

VIRTUAL MACHINE INITIALIZATION AND TERMINATION

Attaching a Virtual Machine to the System

DMKCNSIN
Entered via interruption from a console or terminal (not displays) device. If appropriate, determine and store device type in the RDEVBLOK. Write the VM/370 online message. Sets up to receive attention interruption.

DMKBLDVM
On attention interruption, build skeleton VMBLOK for LOGONxxx.

DMKCFMBK
Send read CCWs to the terminal for LOGON or DIAL response.

DMKTRMID
On response determine translate tables to be used.

DMKCFMBK
Validate command and transfer to DMKLOGON.

DMKLOGON
LOGON command execution.

DMKDIAL
Dial access linkage to multiaccess system.

DMKUDR
Via user directory access, validate user logon eligibility. On acceptance of eligibility, that is the successful completion of logon, build and allocate control blocks and linkages for the user's virtual machine.

IPL the Virtual Machine

DMKCFGIP
For the IPL of a named saved system, the name is verified and resources are checked for availability. Virtual storage is set up with the saved system via SWAPTABLE, SEGTABLE, SHRTABLE updates. For the IPL of device address, the IPL simulator is loaded in the user's storage.

DMKVMIPL
User's page 0, set console address, IPL device address, VMBLOK flags IPL device type and class and user CAW. Read in 24 bytes from the CTCA, reader, DASD or tape unit into the user's virtual location zero. The CCW pointer is now set to the IPLCCW at virtual location X'8' and the program is loaded.

DMKVMI, IPLDONE
For IPL STOP, the virtual machine is placed in console function mode to allow change to nucleus name and apparent storage size before continuation.

DMKVMI, LOADNOW
IPL address is inserted in X'02' if BC mode, or X'BA', if EC mode. The user's CAW and registers are restored and control is given to the user by loading the current PSW at virtual location 0.

DMKUSOLG
Entry is the result of user invoking LOGOFF. Set flags in VMBLOK indicating logout operation.

DMKUSO, US006
Retain line communication, if HOLD operand specified.

DMKUSO, US008
Adjust return address to not run the user.

DMKUSOFF
Set VMBLOK flags.

DMKTRCND
Called to reset tracing.

DMKPERT
Called to reset tracing.

DMKACOTM
Accounting called to compute the connect time for the LOGOFF message.

DMKQCNWT
Write the message to the user.

DMKSCHDL
Called to alter user dispatch status.

DMKCFPRR, DMKCSPO
Reset the virtual machine.

DMKVMCAN
Release or return VMCBLOKs if VMCF is active.

DMKVATBC
Release shadow tables (if any).

DMKSCHRT
Dequeue clock comparator request (if any).

DMKBLDRL
Release segment tables, page and swap tables related to the user.

DMKUSO, US094
Via DMKFRET return user VMBLOKs to free storage.

DMKUSO, US093
For the system operator, clear and reinitialize the VMBLOK.

DMKFRET
Return all other virtual machine control blocks to free storage.

DMKACOFF
Punch an accounting card for the user.

DMKUSO, US098
Free LOGOFF message area. Exit to do free storage maintenance. Exit to DMKCFM or DMKDSPCH.

DMKUSOFL
Entry is the result of the invoked FORCE command.

DMKSCNAU
Locate userid VMBLOK.

DMKUSOFL
Set VMKILL in VMBLOK, build CPEXBLOK and stack it for dispatcher.

DMKDSPCH
Upon CPEXBLOK execution, process as at LOGOFF entry DMKUSOFF.

DMKUSODS
Entry from an invoked CP DISCONN command. Set disconnected VMDISCK in VMOSTAT.

DMKQCNWT
Send disconnect message to user.

DMKUSODS
Increment return address to DMKCFM by 4 to prevent a return read to the user's terminal. Clear VMTERM field to indicate the user terminal is disconnected.

DMKQCNWT
Send message to system operator informing him of user disconnect status. Exit to DMKCFM.

DMKCFMBK
Entry used when the ATTENTION key (or equivalent) is pressed once or twice (according to the VM or CP status) to allow the user to direct a line of input data for CP command processing. Set VMFCWAIT and VMCF bits in VMBLOK indicating wait state and console function mode.

DMKFREE
Builds an 18-doubleword CONBUF buffer for the read operation.

DMKSCNFD
Matches the 8-byte command name against the table of matching command names, the truncations of command names, and the allowable abbreviations, starting at COMNBEG0. The format of the table entry is:

Field	Number of Bytes
Command name	8
Class mask	2
Abbreviation count	2
Routine address	4

DMKCFM, CONFFIND
After a command match has been made, the privilege class of the command is matched with the user's privilege class, VMCLEVEL in the VMBLOK.

DMKCFM, CONFCALL
The last four bytes of a command contain the address of the routine that processes the command.

See "CP Diagnostic Aids" for a list of all CP commands and the associated processing modules.

DMKQCNRD
Read in the terminal input command line.

DMKCFMAT
On NULL data and ATTN key indication, post attention interrupt pending in VDEVBLOK, VCUBLOK and VCHBLOK. Return to run the virtual machine.

DMKCFMRQ
On receipt of CP commands ATTN or REQUEST, process the same as previous entry, DMKCFMAT.

DMKCFM
On receipt of * (asterisk) return to DMKCFMBK to set up another read. If console spooling is enabled, all console input and output including comments are spooled for printer output.

DMKCFMBE
On receipt of BEGIN, simulate the start button on the virtual machine (If optional address is supplied with BEGIN command the supplied address is substituted for the location counter address).

DMKCVTHB
Convert this address to binary notation.

DMKCFMSL
On receipt of the SLEEP command or SLEEP with time value (simulation of virtual machine stop button depression) the VMBLOKs VMSLEEP bit is set. The terminal console keyboard is now inactive until the user hits an ATTENTION key or the SLEEP command times out.

Fast Reflection for the Dispatched Virtual Machine

DMKDSPA
Entry for fast reflection activity. If the user is no longer runnable, or if the system is extending, the fast reflect path is not continued and processing continues at the main dispatcher entry point.

DMKDSP, UPVIRT
If the user is running virtual timers, update and test the user's virtual timers.

DMKDSPA1
If the user is still dispatchable, build the new RUNPSW from either IOOPSW or PROPSW and redispatch the virtual machine.

PSW Validation

DMKDSPB
Entry to dispatcher when the user's PSW has been external to DMKDSP.

DMKDSP, CKPSW
Verify the PSW change.

DMKDSP, CKPEND
Unstack any pending interrupts for the user (if enabled).

MAIN Dispatch Entry

DMKDSPCH
Normal dispatch entry after each interrupt handler has finished processing, and after each CPEXBLOK, I/O request and external interrupt has been serviced.

DMKDSP, RUNTIME
If CPSTATUS indicates return from running a user (CPRUN on), first ensure that supervisor time is being charged to RUNUSER. Check the user for time-slice end or queue-slice end, store the time remaining in the time-slice, and update processor problem state time. Also update virtual timers if running.

DMKDSP, WAITIME
If CPSTATUS indicates return from wait (CPWAIT on), first ensure that supervisor time is being charged to the system. Determine the type of wait (I/O wait, page wait, or idle wait) and save the appropriate new wait time value.

DMKDSP, UNSTACK
For nonrunnable virtual machine, go to label CHKILL in DMKDSP.

DMKDSP, UNSTACK
For runnable user, check pending interruptions for the following:

- DMKDSP, CKPEND
  Per interruption (VMPERPND)
  Pseudo page faults (VMPGPND)
  External interruptions (VMPXINT)
- DMKDSP, UNSTIO
  I/O interruptions (VMIOINT)
- DMKDSP, STORECSW
  I/O interruptions are reflected by swapping user PSWs and storing the unit address and status in low storage.
- DMKDSP, CLEARVMX
  Clear the pending bits in the VMBLOK.

DMKDSP, CKPSW
Validate the PSW.

- DMKVATBC
  For virtual machine leaving EC mode, clean up the shadow tables.
- DMKVATMD
  For virtual machine in BC mode and entering translate mode, initialize shadow tables.

DMKDSP, DSPERMSG
For PSW invalid, send error message to virtual machine, and place user in CP mode. If disconnected and invalid PSW, log off user.

DMKDSP, DISPATCH
Complete processing for current user. Call DMKSCHDL if necessary to alter user's dispatching priority.

Selecting the Next Unit of Work

DMKDSP, CKCPSTAK
Process a stacked request. First check the stack of IOBLOKs and TRQBLOKs. If system is not extending, unstack normally. Otherwise, only unstack paging or PCI IOBLOKs.

DMKDSP, WINDOW
Before examining the stack of CPEXBLOKs, open a window for interrupts if the system is not extending.

DMKDSP, CKCPREQ
Check the stack of CPEXBLOKs. If the system is extending, only unstack those blocks that will allow the extend to complete. If the system is not extending, unstack normally. If a CPEXBLOK for the other processor is encountered, give up the system lock and signal the other processor.

DMKDSP, CKUSERS
If no stacked requests can be unstacked, select a user for dispatching. If the system is locked for running users (such as during extend), load a wait state. Scan the run list for a dispatchable candidate. If none is found, load a wait state. If there is also a runnable user for the other processor, signal the other processor. If a runnable user is found, set up to dispatch this user.

Scheduling Users for Execution

DMKSCHDL
Main entry to maintain queues of runnable and eligible users and to alter the user's dispatching status and (when necessary) his dispatching priority.

DMKSCH, CKRSTAT
If the user is now not runnable, but was runnable before, mark the user as not runnable. If the user is in the eligible list, drop him from the list. If the user is in an idle wait state, drop him from the queue.

DMKSCH, CKRUN
If the user is now runnable, mark him as runnable. If the user was not in Q before, add him to the eligible list.

DMKSCH, CKWAITNG
Look through the eligible list for runnable users to add to active queues.

Other Scheduler Function

DMKSCHST
Set a clock comparator interrupt request.
DMKSCHRT
Reset a clock comparator interrupt request.
DMKSCHMD
Set up a request block for midnight date change.
DMKSCH80
Process a real interrupt timer request.
DMKSCHCP
Process a real CPU timer interrupt.

SPOOLING VIRTUAL DEVICE TO REAL DEVICE

Processing Virtual Output Files

DMKVSPEX
Entry from DMKVIO to initiate SIO on a spooling device that is available (not busy and no interruptions pending).
DMKVSP, OPEN
Determine if output device needs to be opened.
DMKSPLOV
If yes, build message control blocks: SFBLOK and VSPCTLBLOK.
DMKPGTVG
Obtain a virtual buffer; the address is stored in VSPVAGE.
DMKPGTSG
Obtain a DASD page; the address is stored in VSPDPAGE.
DMKVSP, BUILDCTL
Assign a spoolid and the other user, record, and device values plus DMKCVTDT.
DMKCVTDT
Assigns the time stamp and date and stores it in SFBLOK.
DMKVSP, PRTCONT
Generate TAG record at the start of the spool data buffer.
DMKVSP, CCWOK
After CCW validity check, data and CCWs (if appropriate) are moved to the work buffer. Trailing blanks are truncated and when the buffer is full, it is written out to the DASD slot.
DMKVSPVP
On console spooling, the following occurs:
1. Skip to channel 1 every 60 lines.
2. Write out the system console, spool file buffer every 16 lines.
3. Place the system console in a pseudo closed state for checkpoint recovery in the event of system failure.

DMKVSP, LASTCCW
When all CCWs are processed, post interruption pending to the VDEVBLOK, VDEVCSW and return control to the user.

Closing Virtual Output Files

DMKVSFCO
Entry via CP CLOSE command. If device busy, defer close operation by building CPEXBLOK, stack it and exit to dispatcher.
DMKVSP, PRTEOF
On device not busy, write final buffer page to DASD storage.

DMKSPLCV
Queue closed virtual printer or punch spool file to the real spool output device, or transfer the file to another user's virtual reader. Also update the SFBLOK with number of copies printed/punched, distribution code, hold status, and file owner ID. If VSPXBLOK with TAG data exists for the spool device, copy the TAG data to the TAG record in the first spool file data buffer.

DMKSPL, TXTXFR
If a "spooled to" file, queue to the end of the reader file chain. Otherwise, chain the SFBLOK to the designated real spool printer or punch.

DMKCKSPL
Checkpoint the new spool file block.

DMKSPL, SETPEND
For a "spooled to" file find a virtual reader with the proper class and in the ready state with no active file, and no pending interrupts. Then build an IOBLOK with IOBIRA of DMKVIOIN.

DMKSTKIO
Stack the IOBLOK.

DMKSPL, SETPEND
Exit to DMKVSP.

DMKSPL, TSTHOLD
For not "spooled to" files and not in user or system hold, find printer or punch with the proper class. Then build an IOBLOK with IOBIRA of DMKRSPEX.

DMKSTKIO
Stack the IOBLOK.

DMKSPL, TSTHOLD
Exit to DMKVSP.

Processing Virtual Input Files

DMKVSP, OPENRDR
Entry to open a spool input file. If VDEVSPL=0 the file needs to be opened. Build VSPLCTL block and a work buffer. Search the system reader file chain per PSA linkage ARSPRD for a file with appropriate user and class.

DMKVSP, SETFLAG
On file-found condition, place first DASD page address in VSPLCTL, VSPDPAGE. Obtain a virtual buffer and retain its address in the VSPLCTL block.

DMKVSP, READER
Check the CCWs for validity, move and expand the data back to its original size and the data is moved from the work buffer to user's virtual storage.

DMKVSP, RDRCOUNT
On EOF, set SFBEOF bit in SFBLOK and return to caller.

Closing Virtual Input Files

DMKVSPCR
For CLOSE operation requested via console command and the device is busy, initiate a delayed close by constructing and stacking the CPEXBLOK for the CLOSE.

DMKVSP, RDREOF
For normal end of file and VDEVSFLG indicates continuous read.

DMKVSP, OPENCONT
Locate the next file and continue reading.

DMKVSP, LASTFILE
For last file, post end status in RDEVBLOK.
DMKVSP, FILECLR
For HOLD status file (VDEVSFLG=VDEVHOLD), call DMKCKSPL.
DMKCKSPL
Checkpoints the file.
DMKVSP, FILECLR
Unchain the file (except hold files) from the reader queue and call DMKSPLDL.
DMKSPLDL
Delete the file.
DMKVSP, DVICECLR
To clear the device, call DMKRPAGT.
DMKRPAGT
Releases the storage page.
DMKPGTVR
Releases the virtual buffer.
DMKFRET
Releases storage for the work buffer and VSPLCTL block.

SPOOLING TO THE REAL PRINTER/PUNCH OUTPUT DEVICE

DMKRSPEX
Entry from the dispatcher when an IOBLOK is unstacked with and interrupted for spooling unit record device. IOBRADD points to the RDEVBLOK RDEVTYPC input or output class.
DMKRSP, RSPLOUT
If RDEVSPOL indicates an available spool device (not active),
DMKFREE
Get storage for a work buffer and build a RSPLCTL block and link it to RDEVBLOK.
DMKRSP, PRNXTFIL
Search printer and punch SFBLOK chains for corresponding device and class. On a found condition, unchain the block, put its address in RSPSFBLK. The FLASH name specified in the SPOOL command, if FLASH is specified, must match the flash overlay name for a 3800 printer.
DMKSEPSP
If called, provides separators for output pages or cards.
DMKTCSET
If the device is a 3800 printer, call this module to set it up.
DMKRSP, PROCESS1
Bring first spool data DASD page to the work buffer and convert CCW addresses to real device addresses.
DMKIOSQR
Start the spool device.
DMKRSP, PRNXTPAG
Repeat the process until done.
DMKRSP, REPEAT
Reprocess and reaccess the buffer, if multiple copies are specified.
DMKCKSPL
Checkpoint records the change to COPY count.
DMKSPLDL
Delete the file on completion (unless HOLD specified). If the device is a 3800 printer, check for delayed purge.
DMKRSP, PRNXTFIL
Locate the next spool file to process.
DMKRSP, PRTIDLE
Processing for the device is complete as there are no more SFBLOK, for this device or the device was drained.
DMKFRET
Release work area and completed IOBLOK storage.
DMKDSPCH
Exit to the dispatcher.

SPOOLING TO THE REAL INPUT DEVICE

DMKSPLOR
Assume there is no active file being processed on the real input file reader. The spooling operator has issued the START command to the device to "open" the reader.

DMKSPL, BUILDCTL
Build RSPLCTL and SFBLOK.

DMKPGTVG
Get virtual buffer and place its address in RSPVPAGE.

DMKPGTSG
Get DASD buffer and place its address in SFBSTART and RSPDPAGE, linke together by pointers.

DMKIOSQR
Start the reader.

DMKDSPCH
Await the interruption.

DMKRSP, RDERGETID
Check that the first card in the buffer is the userid header. If so, proceed.

DMKRSP RDRCARDS
Preload the buffer with CCWs.

DMKIOSQR
Issue the SIO (SIO's of 42 cards per buffer load).

DMKRSP, RDRSIO
Write the buffer to the DASD slot. Repeat until EOF detected.

DMKSPLCR
Close the file on EOF. Queue the file on reader spool chains.

DMKCKSPL
Add the spool reader file block to the checkpoint cylinder data.

DMKSPL, RDRPEND
If the file owner is logged on, and his virtual reader is available, an IOBLOK is constructed with device end pending -

DMKSTKIO
Stacks it.

DMKRSP, RDREXIT4
Release storage for virtual buffer, RSPLCTL and the SFBLOK.

DMKDSPCH
Exit to the dispatcher.

SPOOL FILE DELETION

DMKPLDL
With R7 not equal to zero, place the specified SFBLOK on the delete chain anchored to DMKRSPDL.

DMKCKSPL
Delete the SFBLOK from checkpoint cylinder data.

DMKSPLDL
Assume the delete routine is not running, build a CPEXBLOK to call DMKSPLDR.

DMKSPLDR
Sets the DELSW=X'80' (delete routine active).

DMKSTKCP
Stacks it and exits to caller.

DMKSPLDR
On unstacking the CPEXBLOK, if the SFBLOK is a system dump file, calls DMKDRDDD.

DMKDRDDD
Deallocates DASD buffers.

DMKSPL, NEXTSFB
For complete allocation chains of RECBLOKS, call DMKPGTSR

DMKPGTSR
deallocate DASD buffer and return to storage held by the dummy RECBLOKs.

DMKSPL, DELSTART
For incomplete allocation RECBLOK chains, deallocate by calling DMKPGTSD.

DMKPGTSD
Deallocates a page at a time via SFBSTART and the IOBLOK until the last page is reached.

DMKFRFT
Delete the SFBLOK, then go to DMKSPL and NEXTSFB.

DMKSPL, NEXTSFB
If the delete queue is not empty, process the next SFBLOK an identical manner. Continue until all SFBLOK deletions are complete then call DMKFRET.

DMKFRET
Delete the IOBLOK.

DMKDSPCH
Exit to the dispatcher.

RECOVERY MANAGEMENT SUPPORT OPERATION

Establishing the Error Recording Base

DMKIOEFL
Entry from CP initialization module to set up pointers to VM/370 error recording cylinders.

DMKIOGF1
The STIDP instruction stores processor version and model in CPUID of PSA.

DMKIOG, ISSUEINS
Check attached channels. If standalone channel on the 165 or 168, the address of the logout routines is stored in the DMKCCH module.

DMKIOG, CHANGEID
Set up pointers for machine check and channel check record area and extended logout areas.

DMKIOG, IOGMCHIN
Obtain storage for machine check record, extended logout area, and CPEXBLOK. The MCHAREA is also initialized.

DMKIOG, PASTDAVE
Determine the 90%-full and 100%-full capacity of designated error recording cylinders and store the amount in DMKIOEMX and DMKIOENI respectively.

DMKIOG, FINDREC
Check first record of the error recording cylinders for proper format. If invalid, reformat. If valid but clear, store pointer value in PSA as the first available slot for error record. If valid but used, search for first unused slot and store its value in PSA.

DMKIOGFR
If on a 3031, 3032, or 3033 processor, read frames from the SRF (service record file) device, and write them to the beginning of the error recording cylinders with unique record types.

DMKIOG, CYLFULL
When error recording area is full, inform the operator, and continue.

DMKIOEFL
Turn off the recording in progress switch and exit to caller.

Process the Machine Check Interruption

DMKMCHIN
Entry via the machine check PSW upon detection of an unrecoverable and nonfatal processor or storage error. Disable soft machine recording store logout area on the machine check and channel check recording cylinders. The system is enabled for hard machine checks with a pointer to the termination routine. DMKMCH, ENHARD for virtual user store status in VMBLOK. DMKMCH, MCHSYSIL for system damage timing facility or uncorrectable retry, multibit storage error post system operator message, flag system as terminated. Place wait state code, if first hard error, record it. If the fault occurred in problem state, terminate the active virtual machine.

DMKMCH, SOFTSTG
For corrected ECC or processor retry, update soft error count and record the error and dispatch the virtual machine.

DMKMCH, MCHSKIP
For multibit storage error in problem mode, exercise storage location to clear up or flag as unavailable (permanent error).

DMKMCH, MCHCHANG
On an altered page condition, the virtual machine is reset, otherwise, the error is recorded and the virtual machine is redispatched.

DMKMCH SPFTEST
Storage key failure. Exercise the 2K page key. If CP area and solid error condition process as DMKMCH, MCHSYSIL, intermittent, restore the key and go to the dispatcher. If key failure and in virtual machine area if permanent error, mark page as unavailable, terminate the user. If intermittent condition refresh the key and dispatch the virtual machine.

DMKMCH, VIRTERM
On conditions that cause the termination or reset. The error is recorded, and both the user and the operator receive status messages. Per the termination flag, VMBLOK, the user is logged off and control returns to the dispatcher or is reset via DMKCFPRR.

DMKCFPRR
Virtual storage is released, the virtual machine is flagged dispatchable and placed in console function mode.

DMKMCH, TERM
On a hard machine check while handling a machine check, the machine check new PSW is loaded with a wait state PSW and the current PSW is enabled for hard machine checks.

DMKMCH, MCHTERM2
Locate the system or the user's VMBLOK.

DMKMCH, OPCOM
Call DMKMCTPT if system is running in attached processor mode.

DMKMCH, MCHWAIT
Load disabled wait state for uniprocessor system.

DMKMCTPT
Complete processor termination for attached processor system. If the error is on the attached processor and it is in problem state, signal for automatic processor recovery and stop the attached processor.

DMKMCT, SWITCH
Make sure processing is on the main processor and set up the appropriate wait state code.

DMKMCT, OPCOM
Issue a message to the operator and load a disabled wait state for the attached processor system.

DMKMCTPR
Perform automatic processor recovery function. Allow system to convert to uniprocessor mode by calling DMKCPUUP.

DMKMCT, PREXIT
Terminate the virtual machine if it is in control. Reset the main processor timer. Clear all lock words and return to the dispatcher.

Process the Channel Check Interruption

DMKCCHIS
Entry via DMKIOS via CSW channel error
DMKFREE
Obtain storage and build a CCHREC block and if IOBLOK and RDEVBLOK exist, build an IOERBLOK.
DMKCCH, CCHIOERL
Store the CCHREC address, its length, and the CSW in the IOERBLOK.
DMKCCH, CCHDEPND
Call appropriate channel error analysis module. Analyze channel logout data for validity.
DMKCCH, SCNEND
Record the error on the error recording cylinder, if appropriate.
DMKCCH, CPTERM
Terminate CP if the PSA's terminate flag is set.
DMKCCH, CCHWAIT
Set up X'0F' wait state code and call DMKMCHST to terminate the system.
DMKMCHST
If the system is running in attached processor mode, call DMKMCTST.
DMKMCH, CALLOPR
Issue an error message to the operator.
DMKMCH, MCHWAIT
Load a disabled wait state for a uniprocessor system.
DMKMCTST
Make sure system is running on the main processor.
DMKMCT, CALLOPR
Issue an error message to the operator.
DMKMCT, MFAWAIT
Load a disabled wait state for attached processor system.
DMKCCH, SCNEND
Unless termination is established, return to DMKIOS for recovery.

Recording the Errors of the Virtual User Via SVC 76

DMKVERD
Entry via DMSPSA as a result of SVC 76 detection. Check parameters passed in R0 and R1.
DMKFREE
Obtain storage for a record buffer for the user error record.
DMKVER, BUFFUL
Using valid record type (from the buffer) branch to an appropriate routine to format that particular record type.
DMKVER, VER30
Using RDEVBLOK, VDEVBLOK and VMBLOK, convert virtual data to real values and place in record.
DMKIOERV
Record the error.
DMKDSPCH
Exit to dispatcher.

DMKUDRFU
Entry after CP detected LOGON command. DMKSYSPL points to the directory. Determine length of userid, if valid call DMKLOCKQ.

DMKLOCKQ
Lock the directory in storage.

DMKUDR, NXTPAGE
Bring in each directory page and return each page (and clear the buffer) until a UDIRBLOK match occurs or directory's last page is detected.

DMKUDR, FINDUSER
On userid found, move UDIRBLOK to caller's area.

DMKLOCKQ
Unlock the directory in storage.

DMKUDR, EXITCC0
Return to caller.

DMKUDRFD
Entry from calling routine to find the addressed (cuu) device UDEVBLOK in users directory and move it to the caller. Via UMACBLOK locate the UDEVBLOKs.

DMKUDR, FINDDEV
Check to see if the user device address is the same as in the UDEVBLOK. Search the chain until match or end of chain occurs.

DMKUDR, DEVFOUND
For found condition, post condition code zero in user's VMPSW.

DMKUDRRD
Entry from calling routine to read the UDEVBLOK addressed into the caller's buffer. Using the DASD and the user displacement from the UMACBLOK, bring in the buffer page to storage. Determine if the virtual directory page address (UDBFVADD) exists in the user directory buffer blocks. If not call-

DMKPGTVG
and get a virtual page.

DMKRPAGT
For DASD address does not match the UMACBLOK, point to the DASD page and bring in the virtual buffer page. Move UDEVBLOK into callers area and set cc=0 in VMPSW. Return to caller.

DMKUDRRV
Entry to return a virtual page used as a buffer. Determine if UDBFBLOK contains a virtual buffer page pointer (UDBFVADD). If not, exit with cc=1 set in the VMPSW. If a buffer exists, check to see if it is resident; if it is, clear it to zeros.

DMKPAGT
Return the real page to the system.

DMKRGTVR
Return the virtual page to the system.

DMKUDRRV
Set cc=0 and return to caller.

DMKUDRBV
Entry from DMKDIRCT or DMKCPINT to build page buffers for each UDIRBLOK.

DMKFREE
Get storage for the virtual buffer page list.

DMKUDR, GETVPAGE
Call DMKPGTVG and DMKRPAGT to get the virtual and real buffer. Save the virtual buffer address in the page list.

DMKUDR, FRETLIST
Encountered I/O error, free the virtual buffer page list, post fatal message, set cc=3 and return to caller.

DMKUDR, ENDLIST
Swap the new virtual buffer page list with the old list. Anchor the new list to DMKSYSPL.

DMKUDR, FRETLIST
If there was a previous buffer page list, free it. Save the start of the user directory pointer in DMKSYSUD, and return to caller with a cc=0 in the VMPSW.

SAVE THE 3704/3705 CONTROL PROGRAM IMAGE PROCESS

DMKSNCP
Entry from DMKHVC and DIAGNOSE code 50. Per the system VMBLOK, locate the DMKRNTBL. The CCPARM virtual address is contained in R1 of the DIAGNOSE instruction.

DMKSNC, NAMECHK
Match via search CCPARM; CCPNAME with DMKRNTBL entries.

DMKSNC, SIZECHK
Verify DASD space requirements for 3704/3705 control program and resource data. The volume required to save (NCPVOL) as indicated in the NCPTBL entry must be available and mounted on the system, on a CP owned and supported paging device.

DMKSNC, SVRESDAT
Save resource data on the NCPVOL device. CCPARM supplies the starting address and size parameters for this write operation.

DMKSNC, SVNCPIM
Save 3704/3705 control program image on NCPVOL device. CCPARM also provides the parameters for this similar operation.

DMKSNC, SAVEFINI
Store cc=0 on no errors and return to caller.

SPOOL FILE CHECKPOINT AND RECOVERY

Initialization

DMKCKSIN
Entry from CP initializer, DMKCPI to initialize the checkpoint cylinders. Per DMKSYSCH, get a virtual page for the checkpoint cylinder and set up the device code in the system residence device. In addition, set up local data areas such as pages per cylinder and checkpoint cylinders.

DMKCKS, CKSIN1
Loop through each SFBLOK in the system and checkpoint it in a slot on the checkpoint cylinder. Then loop through each remaining slot and mark it empty.

DMKCKS, CKSINS
Place the map delimiter of the last non-empty slot in the map.

DMKPTRUL
Unlock the map page.

DMKCKS, CKSIN5
Return to caller.

Dynamic Checkpoint of Spool Files and Spool Devices

DMKCKSPL
Entry from any routine that adds, deletes, changes, the status of closed spool files. Lock the routine, or wait until it becomes unlocked. Bring the map page into storage and set up the device code of the system residence volume.

DMKCKS, LOOPSHQ
If the change is applicable to a SHQBLOK (hold queue block), make appropriate change on the checkpoint cylinder.

DMKCKS, CKSPL1
If the change is applicable to a SFBLOK, either add, change, or delete it on the checkpoint cylinder.

DMKCKS, CKSPL5
If the change affects a spooling device RDEVBLOK (for example, a START or DRAIN command issued), mark the change on the checkpoint cylinder.

DMKCKS, CKSEXIT
Unlock the routine. Unlock the page map and exit to caller.

Reconstruction of Checkpointed Closed Spool Files

DMKCKSWM
Entry via DMKCPI during VM/370 reinitialization process whenever the records for closed spool data need to be reconstructed. Get a virtual page for the map of the checkpoint cylinder and set up the device code of the system residence volume. In addition, set up local data areas.

DMKCKS, CKSWM2B
For slots having real device entries, set or reset the RDEVDISA and RDEVDRAN and move in the checkpointed device classes into RVDEVCLAS.

DMKCKS, CKSWM2G
For slots containing spool hold queue block, chain this to the SHQ chain.

DMKCKS, CKSWM3
Get storage for SFBLOK space and set flags depending upon its last checkpoint activity.

DMKCKS, CKSWM4
If the file SFBLOK was active, chain it to the appropriate printer, reader, or punch chain.

DMKCKS, CKSWM5
Allocate the DASD buffers of the spool file by reading each buffer to determine the next one and then allocate this page.

DMKCKS, CKSWM6E
For the dump spool file, the buffers are allocated sequentially from the beginning to the end.

DMKCKS, CKSWM9
Set up the map delimiter for the end of non-empty slot; then set up a new spool file identity (spoolid) higher than existing numbers. Return to DMKWRM.

INTER-VIRTUAL MACHINE COMMUNICATION

DMKVMCFC
Entry from DMKHVC and the DIAGNOSE instruction code X'68'. Builds a VMCBLOK and initializes it with data from the user's parameter list, VMCPARM. The virtual address of VMCPARM is contained in bits 8-11 (rx) of the DIAGNOSE instruction.

DMKVMC, VMCFTBL
Branch table to pass control to the appropriate subroutine based on the subfunction code in VMCPARM.

Subfunction Code	Subroutines
X'0000'	VMCAUTH
X'0001'	VMCUAUTH
X'0002'	VMCSEND
X'0003'	VMCSENDR
X'0004'	VMCSENDX
X'0005'	VMCRECV
X'0006'	VMCCNCL
X'0007'	VMCREPLY
X'0008'	VMCQIES
X'0009'	VMCRESUM
X'000A'	VMCIDENT

DMKVMC, VMCWAKUP

Notifies a virtual machine of a pending VMCF communication by posting a special external interrupt X'4001' unless:

- There is already a special external interrupt posted.
- The virtual machine is running disabled for VMCF interrupts (PSW bit 7 and CR0 bit 31).

DMKVMC, VMCXFER

Transfers data from one virtual storage to another virtual storage. Errors occurring during data transfer are reflected to originating virtual machine via the data transfer return code in the final response interrupt message header.

DMKVMCEX

Called from DMKDSP to reflect an external interrupt message header to a virtual machine. If the VMCF subfunction is a SENDX, the SOURCE data is moved into the external interrupt buffer immediately following the message header.

DMKVMCUA

Called by DMKCFP when a virtual machine is logged off or reset. Uses the VMCUAUTH subroutine (subfunction code X'0001') to dispose of existing VMCBLOKS before turning off virtual machine communication.

CP Directories

This part contains the following directories:

- CP Module Entry Point Directory
- CP Module-to-Label Cross-Reference
- CP Label-to-Module Cross-Reference

CP Module Entry Point Directory

Module Name	Entry Points	Attributes, Function
DMKACO		Pageable.
	DMKACODV	Builds an account card buffer for a VDEVBLOK.
	DMKACOFF	Creates account card buffer for a VMBLOK.
	DMKACON	Provides additional accounting function at logon time (for installation use).
	DMKACOPU	Punches queued up accounting cards.
	DMKACOQU	Queues up account card buffers for output on a real device.
	DMKACOTM	Creates a connect and usage time message for a user.
DMKALG		Pageable.
	DMKALGON	Handles the AUTOLOG command.
DMKAPI		Pageable. This module is entered from DMKCPI only if in attached processor mode. It is also entered from DMKCPU as part of the vary online processor function.
	DMKAPIPR	Initializes the PSAs for each processor.
	DMKAPIAP	Initializes the control registers for the attached processor.
DMKATS		Pageable.
	DMKATSCF	Notifies the virtual machine that the command has replaced the shared system with a private copy of that shared system. The user continues to run without the shared copy of the named system. Called by the command processors via an SVC if the command execution is to change a shared page.
DMKBLD		Pageable.
	DMKBLDEC	Allocates storage for a virtual ECBLOK and the two TRQBLOKs required for a virtual machine with the ECMODE option, and initializes these blocks.
	DMKBLDRL	Releases real segment, page, and swap tables to free storage.
	DMKBLDRT	Creates and initializes segment, page, and swap tables as a function of virtual storage size, which is part of the process of building a user's virtual machine.
	DMKBLDVM	Creates and partially initializes a VMBLOK for a virtual machine, identified by its terminal real device block.
DMKBOX		Pageable. Provides the VM/370 or user logo (header) for printed output.
	DMKBOXBX	Logo for initial screen display and header separator for printer spool files.
	DMKBOXHR	Installation header reference.

Module Name	Entry Points	Attributes, Function
DMKBSC		Resident. Line error processing for remote 3270s on binary synchronous lines only.
	DMKBSCER	Examines the error condition resulting from a unit check or channel error that occurred while executing a CP-generated bisync line channel program. If the error is uncorrectable, DMKMSW is called to notify the operator. After return from DMKMSW, the original channel program is terminated and the fatal flag is set in the IOBLOK. If the error is correctable, the channel program is re-executed up to a maximum of seven retries.
DMKCCH		Resident. Operates with the I/O interrupt handler to schedule a device-dependent error recovery procedure when a channel data check, control check, or interface control check is detected.
	DMKCCHIS	Entry from DMKIOS when a channel check occurs when storing a CSW after a SIO.
	DMKCCHNT	Entry from DMKIOINT when a channel check occurs on an I/O interrupt.
	DMKCCHRF	Reflects channel check information to the virtual machine.
	DMKCCHRT	Entry from DMKIOE to allow error messages to be printed.
DMKCCW		Resident.
	DMKCCWB1 through DMKCCWB8	CP assist TRANBRNG instruction (E60B).
	DMKCCWGN	CP assist CCWGENRL instruction (E60F).
	DMKCCWL1 through DMKCCWL5	CP assist TRANLOCK instruction (E609).
	DMKCCW1	CP assist DECCW1 instruction (E60C).
	DMKCCW2	CP assist DECCW0 instruction (E604).
	DMKCCWSB	Invokes an internal subroutine (CNTRLSUB) to obtain control bytes (seek data).
	DMKCCWTC	Searches previous (external) RCW chains and resolves the address of the RCW task if found.
	DMKCCWTR	Takes the list of virtual CCWs associated with the user's SIO and translates it into a real CCW list.
DMKCDB		Pageable. Processes DISPLAY, DCP commands.
	DMKCDBDC	Executes the DISPLAY command to display real storage locations.
	DMKCDBDI	Displays virtual storage locations, storage keys, general registers, floating-point registers, PSW, CAW, and CSW at the terminal.

Module Name	Entry Points	Attributes, Function
DMKCDM		Pageable Processes DUMP and DMCP commands.
	DMKCDMDM	Dumps real storage to spooled printer.
	DMKCDMDU	Dumps virtual storage to the spooled printer.
	DMKCDMDM	Dumps the contents of the specified real storage locations on the virtual printer spool file.
	DMKCDMDU	Dumps the contents of the specified virtual storage locations, registers, PSW, and storage keys on the virtual printer spool file.
DMKCDS		Pageable. Processes STORE and STCP commands.
	DMKCDSCP	Stores data into real storage (STCP command).
	DMKCDSTO	Stores data into virtual storage (STORE command).
DMKCFC		Pageable. Gets the address of the routine that processes the CP console function that was requested.
	DMKCFCMD	Processes a CP console function.
	DMKCFCSL	Processes the SLEEP command.
	DMKCFCBE	Processes the BEGIN command.
	DMKCFCQU	Processes the QUERY command.
	DMKCFCRQ	Presents an attention interruption to the virtual machine to simulate a real request key interruption.
DMKCFD		Pageable. Processes LOCATE and ADSTOP commands.
	DMKCFDAD	Stops virtual machine at specified address (ADSTOP command).
	DMKCFDLO	Displays address of real device blocks, or VMBLOK and/or virtual device blocks (LOCATE command).
DMKCFG		Pageable.
	DMKCFGCL	Handles Diagnose code X'64'.
	DMKCFGII	Entry to IPL from LOGON (DMKLOG).
	DMKCFGIP	Entry to IPL from a command line (DMKCFM).
DMKCFH		Pageable.
	DMKCFHSV	Saves a virtual machine's storage space including registers and PSW (SAVED System).
DMKCFM		Resident. Processes DIAGNOSE code 8. It scans the command line and goes to the required module.
	DMKCFMAT	Posts an attention interrupt pending for the virtual machine.
	DMKCFMBK	Puts the terminal in console function (CP) mode (ATTN key pressed twice). Scans the command line and goes to the command handling routine.
	DMKCFMEN	Entered when DIAGNOSE code 8 is executed. Scans the command line and goes to the command handling routine.

Module Name	Entry Points	Attributes, Function
DMKCFO		Pageable.
	DMKCFOEX	Processes Class A, B, C, and F SET commands.
DMKCFP		Pageable.
		Simulates the operator's console for the virtual machine.
	DMKCFPRD	Handles virtual device reset for other CP routines.
	DMKCFPRR	Handles system resets for other CP routines. Resets the virtual machine.
DMKCFS		Pageable.
		Processes the CP SET command for general users.
	DMKCFSET	Entry point for SET command processor.
DMKCFT		Pageable.
		Processes user's terminal options.
	DMKCFTRM	Entry point for the TERMINAL command processor.
DMKCKP		Pageable.
		Saves pertinent data when a check point occurs.
	DMKCKPT	Retrieves accounting data from the VMBLOK, VDEVBLOK, and unpunched accounting cards. It retrieves accounting information for dedicated devices, saves the system log messages, and saves all control blocks for spool files. The data is written on the SYSWARM cylinder of the IPL pack. DMKCKP is loaded and executed by DMKDMP or initial program load.
DMKCKS		Pageable.
		Performs checkpoint processing.
	DMKCKSPL	Performs a checkpoint on any alterations in the spool file set up to allow the recovery routine to get them if warm start fails.
	DMKCKSIN	Initializes the check point cylinder after a successful warm start from the standard recovery procedure or after a cold start.
	DMKCKSWM	Recovers previously checkpointed spool file information. This information includes all open print or punch files in existence at the time the system went down or was shutdown. All open spool files are put in user hold status.
DMKCLK		Pageable.
	DMKCLKCK	Determines if the clock should be synchronized. (Called from DMKCPI)
	DMKCLKCC	Handles CLKCHK signal request.
	DMKCLKMP	Synchronizes the clocks.
	DMKCLKAP	Handles SYNC signal request.
	DMKCLKSC	Handles the TOD-sync-check external interrupt.
DMKCNS		Resident.
		Real console terminal manager.
	DMKCNSED	Edits the input line for the following characters: escape, line end, line delete, and character delete.
	DMKCNSEN	Enables or disables a low-speed terminal line.
	DMKCNSIC	Entered from DMKQCN to initialize read and write CCWs for the CONTASK built by DMKQCN.

Module Name	Entry Points	Attributes, Function
	DMKCNSIN	Interruption return point and handler for terminal I/O.
DMKCPB		Pageable. Simulates the operator's console for the virtual machine.
	DMKCPBEX	Processes the EXTERNAL command to present an external interruption to the virtual machine.
	DMKCPBNR	Processes the NOTREADY command to cause the virtual device to appear not ready.
	DMKCPBRS	Processes the RESET command to reset all pending interrupts from the specified device.
	DMKCPBRW	Processes the REWIND command to issue a rewind to the real tape device.
	DMKCPBRY	Processes the READY command to simulate a device end interrupt to the specified device.
	DMKCPBSR	Processes the SYSTEM command to simulate system reset and PSW restart to allow clearing of storage.
DMKCPE		Resident. Contains data constants that define the end of the CP nucleus.
DMKCPI		Pageable. Prepares VM/370 for operation.
	DMKCPIEM	Enables the operator's console, initializes the TOD clock and directory, allows operator logon, prepares for warm start, and completes initialization.
	DMKCPINT	Initializes and prepares CP for operation.
DMKCPS		Pageable. Processes the SHUTDOWN, HALT, and VARY commands.
	DMKCPSSH	Processes the SHUTDOWN command.
	DMKCPSH	Processes the HALT command.
	DMKCPSRY	Processes the VARY command.
DMKCPU		Pageable.
	DMKCPUVY	Processes the VARY PROCESSOR command.
	DMKCPUUP	Causes the system to convert to uniprocessor mode.
DMKCPV		Pageable.
	DMKCPVAA	Punches user accounting records.
	DMKCPVAC	Processes the ACNT command to create accounting records for logged-on users. Also, resets accumulated accounting information.
	DMKCPVAE	Enables system low-speed lines for system restart.
	DMKCPVDS	Processes the DISABLE command to disable an active line after the current user is finished with it.
	DMKCPVEN	Processes the ENABLE command to enable the system's low-speed lines for system logon.
	DMKCPVLK	Processes the LOCK command to lock specified pages of a user's virtual storage space into real main storage.
	DMKCPVUL	Processes the UNLOCK command to unlock pages that were locked by operator command (LOCK).

Module Name	Entry Points	Attributes, Function
DMKCQG		Pageable. Processes the class G and class D QUERY commands.
	DMKCQGEN	Entry to QUERY command processor for class G users.
DMKCQH		Pageable.
	DMKCQHRD	Processes QUERY RDR command.
	DMKCQHPR	Processes QUERY PRT command.
	DMKCQHPU	Processes QUERY PCH command.
DMKCQP		Pageable. Processes the class B and class G QUERY command.
	DMKCQPRV	Entry to QUERY command processor for class B and G users.
DMKCQR		Pageable. Processes the QUERY command.
	DMKCQREY	Main entry point. Contains a branch table to get to the routine that processes the operand specified in the QUERY command; the operand can be one of the following: FILES, SET, DUMP, PAGING, HOLD, PRIORITY, and TERMINAL.
	DMKCQRFI	Retrieves the number of reader, punch, and print files.
DMKCQY		Pageable.
	DMKCQYEY	Handles QUERY functions: TIME, LOGMSG, NAME, USERS, PF, SASSIST, CPASSIST, and CPUID.
DMKCSB	DMKCSBLD	Processes the LOADBUF command (real UCS or FCB buffer).
	DMKCSBVL	Processes the LOADVFCB (load virtual forms control buffer) command.
DMKCSO		Pageable. Processes real spooling commands for real unit record devices.
	DMKCSOBS	Processes the BACKSPACE command.
	DMKCSODR	Processes the DRAIN command.
	DMKCSOFL	Processes the FLUSH command.
	DMKCSORP	Processes the REPEAT command.
	DMKCSOSD	Restarts a device after it has been drained.
	DMKCSOSP	Processes the SPACE command.
	DMKCSOST	Processes the START command by device type.
DMKCSP		Pageable. Processes class D and G spooling commands.
DMKCSQ		Pageable.
	DMKCSQCL	Processes the CLOSE command.
	DMKCSQHL	Processes the HOLD command.
	DMKCSQRF	Processes the FREE command.
DMKCST		Pageable. Processes class G commands.
	DMKCSTAG	Entry point to process the TAG command.
DMKCSU		Pageable. Processes the class D and G spooling commands.
	DMKCSUCH	Processes the CHANGE command.

Module Name	Entry Points	Attributes, Function
DMKCSV		Pageable.
	DMKCSVOR	Processes the ORDER command.
	DMKCSVPU	Processes the PURGE command.
	DMKCSVTR	Processes the TRANSFER command.
DMKCVT		Resident. Processes the conversion routines.
	DMKCVTBD	Converts a word of binary data into a doubleword of decimal digits.
	DMKCVTBH	Converts a word of binary data into a doubleword of hexadecimal data.
	DMKCVTDB	Converts a decimal field into a fullword of binary data.
	DMKCVTDT	Converts data and time to EBCDIC and inserts it into a specified location.
	DMKCVTFP	Converts a floating-point doubleword into 17 bytes of decimal data.
	DMKCVTHB	Converts the designated hexadecimal field into a binary fullword.
DMKDAS		Resident. DASD error retry program.
	DMKDASER	Retries the failing DASD channel program.
	DMKDASRD	Processes unsolicited device end interruptions.
	DMKDASSD	Collects DASD sense data.
DMKDDR		Residency not applicable. This is the DASD dump restore program. It saves data from a direct access volume onto a tape or tapes. It returns data to DASD from tape that has been placed on the tape by this program. It copies data from one device to another of the same type. It prints a translation of each record specified on the SYSPRINT device. Prints a translation of each record specified on the console. Initial program loaded or run under CMS if on a CMS disk.
	DMKDDREP	DASD dump restore program entry point.
	DMKDDRED	End-of-load module for CMS.
DMKDEF		Pageable. Processes the DEFINE command to define a virtual device or storage.
	DMKDEFIN	Processes the DEFINE command to alter the virtual machine's configuration or storage size.
DMKDGD		Resident. Processes simple disk I/O.
	DMKDGDDK	Performs simple disk I/O of a standardized format with a minimum of CCW chain manipulation and interruption handling.
DMKDIA		Pageable.
	DMKDIACP	COUPLE command processor. Establishes a virtual connection between two channel-to-channel adapters on a single virtual machine.
	DMKDIADR	Releases a terminal line that has been in use by the virtual machine via the DIAL command. The line is detached from the virtual machine and made available for normal log on to VM/370.

Module Name	Entry Points	Attributes, Function
	DMKDIAL	Processes the DIAL command. Attaches a user's terminal as a dedicated device to an existing virtual 270X terminal line in the virtual machine addressed by the command line.
DMKDIB		Pageable
	DMKDIBSM	Simulates sense data and status for virtual I/O to a simulated I/O device (2702 line or CTCA) that has not yet been activated through either the console function DIAL for 2702 lines, or the console function COUPLE for virtual CTCAs.
DMKDIR		Pageable or standalone. Initial program loaded or run under CMS if on a CMS disk.
	DMKDIRCT	Builds a user directory on a system owned volume using pre-allocated cylinders.
	DMKDIRED	End of load module for CMS.
DMKDMP		Resident. Writes a dump of main storage, control registers, floating-point registers, general registers, and clocks to a specified device.
	DMKDMPDK	Writes the dump on the specified device.
	DMKDMPRS	Initial program loads the system over again.
DMKDRD		Pageable. Process spool files
	DMKDRDDD	Delete system dump spool file.
	DMKDRDER	Manipulates input spool files via a DIAGNOSE code X'0014' issued by the virtual machine.
	DMKDRDMP	Reads a system dump spool file via a DIAGNOSE code X'0034' issued by the virtual machine.
	DMKDRDSY	Reads the system symbol table CSECT via a DIAGNOSE code X'0038' issued by the virtual machine.
DMKDSB		Resident. DASD error retry program.
	DMKDSBRD	Processes unsolicited device end interruptions.
	DMKDSBSD	Collects DASD sense data.
DMKDSP		Resident. Entered after each interruption handler is finished processing and after each stacked CPEXBLOK, I/O request, and external interruption has been serviced. It updates the CPU times charged to the user that has received service, updates all virtual timers, and reflects any pending interruptions for which the user is enabled. After the user's status has been updated, the highest-priority runnable user is dispatched.
	DMKDSPA	Immediate redispatch path for virtual machines. The only status update that occurs is for virtual timers.

Module Name	Entry Points	Attributes, Function
	DMKDSPB	Process new virtual PSW and dispatch. Entered if the virtual PSW has been entered outside of DMKDSP.
	DMKDSPCH	Main entry point. Updates timers and dispatches user.
	DMKDSP0	CP assist DSP0 instruction (E60D).
	DMKDSP1	CP assist DSP1 instruction (E607).
	DMKDSP2	CP assist DSP2 instruction (E611).
	DMKDSPE	Processes interrupt from virtual interval timer.
	DMKDSPQS	Nonexecutable; dispatched user's maximum time slice.
	DMKDSPRQ	Queues anchor for IOBLOKs and CPEXBLOKs.
	DMKDSPRU	Entered in attached processor mode when the system lock is not held.
	DMKDSPNP	Number of dynamically assignable page frames now available in the system.
DMKEIG		Pageable. Analyzes the 2880 channel logout and sets appropriate bits in the ECSW field according to the results of this analysis. It moves the channel logout to the channel check record.
DMKEMA		Pageable.
	DMKEMA	Contains the framework of the common error messages that are generated at various places within CP. Module DMKERM references DMKEMA to write error messages that require variable data to be inserted into them. This module contains no executable code and contains all error messages from 0 to 169.
DMKEMB		Pageable.
	DMKEMB	Contains the framework for the common error messages that are generated at various places within CP. The module DMKERM references DMKEMB to write error messages that require variable data to be inserted into them. This module contains no executable code and contains all error messages from 170 to 423.
DMKEMC		Pageable.
	DMKEMC	Contains the framework of the common error messages that are generated at various places within CP. Module DMKERM references DMKEMC to write error messages that require variable data to be inserted into them. This module contains no executable code and contains error messages 424 and up.

Module Name	Entry Points	Attributes, Function
DMKENT		Resident. Meets the CP nucleus residency requirements for TRQBLOK and CPEXBLOK entries to pageable VM Monitor module DMKMIA.
	DMKENTEC	Used to invoke a MONITOR STOP command via a CPEXBLOK.
	DMKENTET	Used to invoke a MONITOR STOP command via a TRQBLOK request.
	DMKENTFI	Used to complete monitor shutdown processing, via CPEXBLOK.
	DMKENTKC	Used to invoke a MONITOR CLOSE command via a CPEXBLOK.
	DMKENTSC	Used to invoke a MONITOR START SPOOL command via a CPEXBLOK.
	DMKENTST	Used to invoke a MONITOR START SPOOL command via a TRQBLOK request.
	DMKENTTI	High frequency I/O status sampling routine, entered every two seconds via TRQBLOK request.
	DMKENT62	DASTAP class 6 code 2 I/O status sampling routine, called from DMKMONTI every 60 seconds to collect the data accumulated by DMKENTTI.
DMKERM		Pageable.
	DMKERMSG	This is the message writer. Locates the requested message and inserts the module ID, message number, and data. It also prints the message.
DMKEXT		Resident
	DMKEXTSP	Handles all SIGNAL actions after CP initialization.
	DMKEXTSL	Entry point for External Interrupt handler
DMKFCB		Pageable.
	DMKFCB	Contains the forms control load buffer images that the LOADBUF command uses to load the forms control buffer in the 3811 control unit for the 3203 or 3211 printer. The LOADVFCB command also uses DMKFCB to load the forms control buffer in the virtual 3203 or 3211 printer.
DMKFMT		Standalone program. Initial program loaded or run under CMS if on a CMS disk. Adapts parameters from the console or IPL device (card reader) and per forms partial or complete formatting, allocating, and labeling of 2314, 2319, 3330, 3340, 3350 and 2305 DASD devices. The FORMAT program also write-checks the surfaces. Bad surfaces are flagged to prevent their use. No alternative tracks are assigned. OS labels are written to be compatible with OS, but labels indicate to OS that no space is left on the DASD device. All input parameters are verified for correctness.

Module Name	Entry Points	Attributes, Function
DMKFRE		Resident. Free storage manager.
	DMKFREE	Gets space from free storage and processes the CP assist FREE instruction (E600).
	DMKFRERC	Special entry point to acquire free storage. If the storage request cannot be satisfied, a condition code of one is returned to the caller.
	DMKFRERS	Returns subpools to free storage chain.
	DMKFRET	Returns space to free storage and processes the CP assist FRET instruction (E601).
	DMKFRETR	Returns space to free storage; does not release pages.
DMKGIO		Pageable. Initializes supervisor operations for tape, unit record, and nonstandard disk I/O operations.
	DMKGIOEX	Checks device validity and initializes I/O operations on tape, unit record, and nonstandard disk I/O programs per supervisor call. This module presents resultant condition code and CSW (if warranted) to the user.
DMKGRF		Resident. Supports local 3270, 3278 Model 2A, and 3066 devices. DMKGRF processes interruptions and CCWs for the devices. The processing includes message handling and screen management.
	DMKGRFIN	Handles the interruption via an IOBLOK.
	DMKGRFEN	Enables or disables the device.
	DMKGRFIC	Starts a CONTASK from DMKQCN.
DMKGRT		Resident. Contains common data area and subroutines for 3270 display support.
	DMKGRTAB	Computes the next tab position and creates the data stream to position the cursor and insert a logical tab character if necessary.
	DMKGRTFM	Brings in the VM/370 logo and initializes buffer and the CCWs to write the logo in DMKGRT and DMKRGB.
DMKGRW		Non-executable. Contains CCWs and data for 3278 Model 2A operator display console.
DMKHVC		Resident.
	DMKHVCAL	Performs services for the virtual machine as requested via the DIAGNOSE instruction. The specific service performed depends on the code in the DIAGNOSE instruction.
DMKHVD		Pageable.
	DMKHVDAL	Performs services for virtual machines as requested by the DIAGNOSE instruction.
DMKIOC	DMKIOCVT	Converts VM/370 device type to OS/VS device type.

Module Name	Entry Points	Attributes, Function
DMKIOE		Resident. This is the error recording module. It receives all requests for error recording and passes control to the proper pageable routine after checking if a recording is in progress. If a previous request for error recording is in progress, the current request is queued on the appropriate queue for recording at a later time. It makes a check to determine if the recording cylinder is full. DMKIOE also interfaces with the pageable module that initializes and erases the error recording cylinders.
	DMKIOECC	Entry for a channel error condition occurring on a SIO in DMKIOS with a response condition code of one.
	DMKIOECH	Entry for a stacked channel recording request from the channel check handler.
	DMKIOECJ	Entry for a stacked channel check recording request from ERP.
	DMKIOEFL	Entry point to locate the starting page record for recording.
	DMKIOEFM	Entry to clear and format the recording area on disk.
	DMKIOEMC	Entry for machine check recording.
	DMKIOEMH	Entry for a stacked machine check request.
	DMKIOENV	Entry for a stacked environmental recording request.
	DMKIOEOB	Entry for a stacked outboard error recording request.
	DMKIOEQQ	Calls to initiate error recording via DMKIOF (no DMKIOE function performed).
	DMKIOERC	Entry for a stacked erase request.
	DMKIOERN	Processes a 3704/3705 and remote 3270 request.
	DMKIOERR	Schedules recording for unit check, channel data check, and hardware environmental counts.
	DMKIOESD	Records 3330 data.
	DMKIOESR	Schedules statistical data recording.
	DMKIOEST	Schedules the update of a statistical data request.
	DMKIOEVR	Processes an SVC 76 request.
DMKIOF		Pageable. Records system and I/O errors on the system disk in predefined error recording cylinders.
	DMKIOFC1	Records channel check error from SIO in DMKIOS when cc=1.
	DMKIOFIN	Initializes pointers to available recording pages at IPL and after an erase has been completed.
	DMKIOFOB	Records OBR and MDR records.
	DMKIOFM1	Records machine checks.
	DMKIOFST	Updates statistical data counters.
	DMKIOFVR	Records errors when requested by SVC 76.

Module Name	Entry Points	Attributes, Function
DMKIOG		Pageable. Called at initialization to locate the error recording device, locate the last outboard error record and system recordings made on the cylinders, and set the in-storage pointers to the correct values. Initialization for RMS functions is performed after first making a test to determine if CP is running under CP. RMS functions are not activated for a virtual CP environment. This module also erases the recording areas.
	DMKIOGFR	For 3031/3032/3033 processors, reads frames from the SRF device, formats them in 4096-byte blocks, and writes the records to the error recording cylinders.
	DMKIOGF1	Contains all function of DMKIOG except erase.
	DMKIOGF2	Erases (1) error records or (2) error records and frame records from the error recording cylinders, depending on input parameters.
DMKIOS		Resident. Schedules requests for virtual machine and program I/O operations, and services all I/O interruptions.
	DMKIOSHA	Halts an active device and drains all interruptions.
	DMKIOSIN	Processes an I/O interruption.
	DMKIOSQR	Schedules CP-generated I/O operation.
	DMKIOSQV	Schedules a virtual machine I/O operation.
	DMKIOSRW	Processes the IOBLOK used for REWIND.
DMKISM		Pageable.
	DMKISMTR	Finds and modifies an ISAM CCW string.
DMKJRL		Pageable.
	DMKJRLQU	Processes the QUERY command.
	DMKJRLSE	Processes the SET JOURNAL command.
	DMKJRLLO	Processes LOGONs with invalid passwords.
	DMKJRLSL	Processes LINKs which are successful.
	DMKJRLIL	Processes LINKs with invalid passwords.
DMKLDOO		Loader - utility program.
	LDRGEN	Loads assembled program modules into storage at locations other than those assigned by the assembler. It completes linkage among the modules and transfers control to one of the loaded modules for execution.
DMKLNK		Pageable.
	DMKEPSWD	Prompts the user to enter a password, types masking characters if appropriate, reads the password from the terminal, and checks it for a match.
	DMKLNKIN	Links to a virtual DASD because of an issued LINK command.
	DMKLNKSB	LINK subroutines.

Module Name	Entry Points	Attributes, Function
DMKLOC		Resident.
	DMKLOCK	Allows a system resource to be marked in use or not available by a unique 8-character name.
	DMKLOCKD	Dequeues a locked name.
	DMKLOCKQ	Queues or locks a name.
	DMKLOCKT	Tests to determine if a name is locked.
DMKLOG		Pageable.
		Logs on a user or operator.
	DMKLOGA	Processes the AUTOLOG command.
	DMKLOGON	Logs on a user.
	DMKLOGOP	Logs on the operator.
DMKLOH		Pageable.
	DMKLOHON	Constructs and sends logon-related messages to a user or to the operator.
DMKLOK		Resident.
	DMKLOK	handles all locking requests when CP is in attached processor mode.
	DMKLOKDF	Processes an obtain, defer lock request.
	DMKLOKPS	Processes all spin lock requests.
	DMKLOKSO	Processes an obtain, defer request for VMBLOK lock.
	DMKLOKSP	Processes an obtain request for a spin lock that previously failed.
	DMKLOKVM	Processes an obtain, defer request for VMBLCK lock.
	DMKLOKVR	Processes a release request for VMBLOK lock.
DMKMCC		Pageable.
	DMKMCCCL	Handles first level MONITOR command processing.
DMKMCD		Pageable.
	DMKMCDIN	Processes MONITOR INTERVAL commands.
	DMKMCDLI	Processes MONITOR LIMIT ccmmands.
	DMKMCDTI	Processes MONITOR TIME commands.
	DMKMCDST	Processes MONITOR STOP commands.
	DMKMCDSE	Processes MONITOR SEEKS commands.
DMKMCH		Resident.
	DMKMCHIN	Processes a machine check interruption.
	DMKMCHMS	Enables or disables soft machine check recording.
DMKMCT		Resident.
		This module is called by the machine check handler in attached processor mode.
	DMKMCTMA	Handles malfunction alert.
	DMKMCTPR	Handles processor recovery.
	DMKMCTPT	Handles processor termination.
	DMKMCTST	Handles system termination.
DMKMIA		Pageable.
		Provides various facilities associated with automatic monitoring using spool files.
	DMKMIACC	Used for MONITOR CLOSE processing.
	DMKMIADL	Used for DMKMCC display function.
	DMKMIAEN	Used to invoke a MONITOR STOP command.
	DMKMIAIN	Used to invoke a MONITOR START command.
	DMKMIAKC	Used to invoke a MONITOR CLOSE command.

Module Name	Entry Points	Attributes, Function
	DMKMIAMU	Generates informational messages for monitor user.
	DMKMIARO	Opens monitor spool file, gets SFB, etc.
	DMKMIAST	Schedules, via TRQBLOKs, requests to start and stop monitor at specific times in the future.
	DMKMIAWO	Writes a monitor data buffer to a spool file buffer.
DMKMID		Pageable.
	DMKMIDNT	Changes the date in the system low storage at midnight and resets the clock comparator for the next midnight occurrence. DMKMID also sends messages to all users about the date change.
DMKMNI		Pageable.
	DMKMNIDK	Constructs spool file header record.
	DMKMNIDS	Displays automatic monitoring information defined by SYSMON macro in DMKSYS.
	DMKMNIFI	Completes monitor shutdown.
	DMKMNISH	Initializes MONITOR shutdown.
	DMKMNISP	Handles monitor processing for SPOOL to USERID parmeters of START command.
	DMKMNITH	Handles monitor tape header processing.
	DMKMNITR	Writes the MONITOR trailor record.
DMKMON		Pageable. Processes commands and requests associated with the MONITOR, including MONITOR CALL interruptions within CP.
	DMKMONIO	Processes tape interruptions returned by DMKIOS.
	DMKMONMI	Processes a MONITOR CALL program interruption.
	DMKMONPR	Gets space for monitor record and manages buffers.
	DMKMON00	Handles PERFORM (class zero) data collection routine.
	DMKMON40	Handles USER (class four) data collection routine.
	DMKMONTI	Handle timer request interruptions.
DMKMSG		Pageable. Transmits messages to logged-on users for the MESSAGE, SMSG, or WARNING commands. Receives and retransmits lines for the ECHO command for the number of times specified.
	DMKMSGEC	ECHO command processor.
	DMKMSGMS	MESSAGE command processor.
	DMKMSGNH	MSGNOH command processor.
	DMKMSGSM	SMSG command processor.
	DMKMSGWN	WARNING command processor.
DMKMSW		Resident.
	DMKMSWR	Allows system communication with the operator for the enhancement of error recovery procedures.
DMKNEM		Pageable.
	DMKNEMOP	Gets a 5-byte mnemonic opcode for a System/370 binary opcode.

Module Name	Entry Points	Attributes, Function
DMKNES		Pageable. Processes NETWORK operands as follows: POLLDLAY SHUTDOWN DISPLAY VARY TRACE
	DMKNESDS	Processes the NETWORK DISPLAY command.
	DMKNESEP	Processes the NETWORK VARY EP command to switch an NCP communication line to EP mode.
	DMKNESHD	Processes the NETWORK SHUTDOWN command.
	DMKNESPL	Processes the NETWORK POLLDLAY command.
	DMKNESTR	Processes the NETWORK TRACE command.
	DMKNESWN	Processes the NETWORK VARY NCP command to switch an EP communication line to NCP mode.
DMKNET		Pageable. Decodes NETWORK command and enables bisync lines.
	DMKNETAE	Enable binary synchronous lines and remote stations.
	DMKNETWK	NETWORK command decoder.
DMKNLD		Pageable.
	DMKNLDMP	Dumps the 3705 network control program.
	DMKNLDR	Loads the 3705 network control program. These routines may be called by a console command from DMKNET or or internally by DMKCPI (for LOAD) or DMKRNH (for DUMP).
DMKNLE		Pageable.
	DMKNLEMP	Dump the 3705 Network Control Program.
DMKOPR		Resident.
	DMKOPRWT	Provides the necessary support for the VM/370 system console. Certain routines within the control program cannot call DMKQCN to issue writes to the system console. This module determines the system's primary console and builds a channel program to handle the requested call.
DMKPAG		Resident.
	DMKPAGIO	Constructs IOBLOKs and schedules the tasks that move virtual storage pages between auxiliary storage and main storage. It also calculates the total system paging load at user-specified intervals.
DMKPER		Pageable.
	DMKPERCH	Sets a return code of zero in R2.
	DMKPERIL	Resets the interruption.
	DMKPERT	Resets program event recording.
DMKPGS		Pageable.
	DMKPGSPO	Release all the pages of a user's virtual storage from the real storage and from auxiliary storage on the paging device.
	DMKPGSPP	Releases a specified part of virtual storage.
	DMKPGSPR	Calls DMKPTRPU to ensure that the user is not in page wait and then releases the address range contained in R1 through R2. It also unlocks any pages that might have been locked.

Module Name	Entry Points	Attributes, Function
DMKPGT		Resident. DASD storage management.
	DMKPGTCG	Allocates contiguous space for a 3704/3705 dump.
	DMKPGTPG	Allocates a page of DASD storage for either virtual storage paging or for spool file page buffers.
	DMKPGTPR	Releases DASD storage used for virtual storage paging.
	DMKPGTSD	Releases one page of DASD storage used for spooling.
	DMKPGTSG	Allocates a page of DASD storage for spooling.
	DMKPGTSR	Releases a group of DASD storage pages used for spooling.
	DMKPGTVG	Allocates a page of virtual storage belonging to the CP paging VMBLOK.
	DMKPGTVR	Releases a virtual storage page.
DMKPRG		Resident.
	DMKPRGIN	Processes a hardware program interruption.
	DMKPRGRF	Reflects an SVC interruption to the virtual machine.
	DMKPRGSM	Simulates a virtual program interruption.
DMKPRV		Resident.
	DMKPRVLG	Simulates a privileged operation.
DMKPSA		Resident.
	DMKPSACG	Charges accumulated time to a virtual machine.
	DMKPSADU	PSW restart processing. Forces an SVC 0 type of dump.
	DMKPSAEX	Processes external interruptions.
	DMKPSAFC	Checks fetch protection per the CAW key.
	DMKPSAFP	Checks for fetch protection violation per PSW key.
	DMKPSAID	Gets virtual address for any instruction.
	DMKPSARR	Gets the virtual address for an RR instruction.
	DMKPSARS	Gets the virtual address for RS, SI, or SS instruction.
	DMKPSARX	Gets the virtual address for an RX instruction.
	DMKPSASC	Checks storage protection per the CAW key.
	DMKPSASP	Checks for a storage protection violation per the PSW key.
DMKPTR		Resident. Manages the inventory of real system pages, provides real storage space for CP functions and for pages of user and CP virtual storage.
	DMKPTRAN	Translates user virtual storage address to a real storage address.
	DMKPTRFR	Gets a page of real storage.
	DMKPTRFT	Releases a page of real storage.
	DMKPTRLK	Locks a page of real storage and processes the CP assist instruction, PTRLK (E602).
	DMKPTRPW	Called to defer execution of system reset functions when user's virtual machine is in page wait.
	DMKPTRUL	Unlocks a page of real storage and processes the CP assist instruction, PTRUL (E603).

Module Name	Entry Points	Attributes, Function
DMKQCN		Resident.
	DMKQCNCL	Clears CONTASK stack and returns all blocks to free storage.
	DMKQCNET	Processes completed CONTASKS for virtual console spooling, return or no return options, and returns the CONTASK blocks to free storage.
	DMKQCNRD	Starts and queues a console read request.
	DMKQCNSY	Synchronizes virtual machine console activity with internal supervisor activity. This is used during a virtual system reset and during the logoff process.
	DMKQCNTO	Disconnects a virtual machine and sets a TOD clock comparator request to log off the virtual machine after a fifteen-minute delay.
	DMKQCNWT	Starts and queues a console write request.
DMKRGA		Resident.
	DMKRGAIN	This is the second-level interruption handler for remote 3270 stations. This module supports the 3270 remote display and printer stations. It processes interruptions and CCWs for the remote stations, including message handling and screen management.
DMKRGB		Resident.
	DMKRGB	Supports the 3270 remote display and printer stations. It processes interruptions and CCWs for the remote stations including message handling and screen management.
	DMKRGBIC	Initializes and schedules CONTASKs.
	DMKRGBEN	Enables and disables bisync lines and remote stations.
DMKRIO		Resident.
	DMKRIO	Exists as a CSECT and defines the machine's configuration. A basic DMKRIO is shipped with VM/370. DMKRIO can be changed at system generation or whenever new machines are added by using the appropriate macros.
DMKRND		Residency not applicable. Invoked via the NCPDUMP command in CMS.
	DMKRND	This is the interface between the VM/370 dump spool file and the OS-SSP dump format program for printing and formatting dumps of the 3704 and 3705 communications controllers.
DMKRNH		Resident.
	DMKRNHIC	Initializes and schedules the CONTASK fields that comprise the 3704 and 3705 Network Control Program transmission header.
	DMKRNHIN	This is the secondary interruption handler for the 3704 and 3705 communication controllers; it is read when operating in NCP or PEP mode.
	DMKRNHND	Schedules control functions for the 3705 or 3704 Network Control Program.

Module Name	Entry Points	Attributes, Function
DMKRPA		Resident. Virtual storage mapping.
	DMKRPAGT	Page-in from DASD to user's virtual storage.
	DMKRPAPT	Page-out to DASD from user's virtual storage.
DMKRSE		Pageable. Real UR device I/O error handler.
	DMKRSERR	Retries and attempts to recover from real unit record device I/O errors.
	DMKRSESD	Collects 3800 sense data.
DMKRSP		Resident. Manages all spooling operations on the real system unit record devices including printing and punching user-created spool files and reading and queueing reader files from the real card reader.
	DMKRSPER	Processes spooling errors (ERP).
	DMKRSPEX	Processes spooling operations. Entered via a GOTO when DMKDSPCH unstacks an IOBLOK with an interruption for the spooling unit record device.
	DMKRSPUR	Formats the active file message for real unit record devices.
DMKSAV		Pageable. DMKSAVNC is entered via an LDT card from DMKLDR. DMKSAVRS is entered via a BALR from DMKCKP. DMKSAV saves and restores a page image count of the CP nucleus on the system residence disk.
	DMKSAVNC	Writes a page image copy of the CP nucleus.
	DMKSAVRS	Restores a page image copy of the CP nucleus.
DMKSCH		Resident. Maintains queues of runnable and eligible users, alters the dispatching status of users, and periodically recalculates the working set size and dispatching priority of users. DMKSCH contains the routines that maintain the system TOD clock comparator request queue and the code that monitors users with abnormal execution.
	DMKSCHAE	Processes the interrupt occurring when the favored execution measurement interval expires.
	DMKSCHBK	From DMKTMR. Calculates problem state time.
	DMKSCHCP	Interruption from real CPU timer.
	DMKSCHDL	Alters a user's dispatching status.
	DMKSCHMD	Interruption for the midnight date change.
	DMKSCHRT	Resets a clock comparator interruption request.
	DMKSCHST	Establishes a clock comparator interruption request.
	DMKSCH80	Interruption for real timer at storage address 80.
DMKSCN		Resident. Scans module.
	DMKSCNAU	Searches the chain of VMBLOKs for one whose userid matches the one pointed to by register one.
	DMKSCNFD	Finds the next field in an input message buffer.

Module Name	Entry Points	Attributes, Function
	DMKSCNLI	Searches the logged-on virtual machines for any links to a specified minidisk. A link is any virtual device whose ADEVBLOK pointer and relocation factor match those specified.
	DMKSCNP	Finds the RCHBLOK and RCUBLOK that represents the next logical path to the device.
	DMKSCNRA	Computes a full real device address (in cuu form) from the RDEVADD, RCUADD, and TCHADD entries in the real device, control unit, and channel blocks.
	DMKSCNRD	Computes a real device address (in CW form), from the RDEVADD, RCUADD, and RCHADD entries in the real device, control unit, and channel blocks.
	DMKSCNRN	Returns the name of the real device to the caller in register 1.
	DMKSCNRU	Returns the addresses of the real channel, control unit, and device blocks for a given real device to to the caller.
	DMKSCNVD	Computes a full virtual device address (in cuu form), plus the addresses of the virtual channel and control unit blocks from a specific virtual device block.
	DMKSCNVN	Returns the name of the virtual device to the caller in R1.
	DMKSCNVS	Searches all the real device blocks for a device whose volume serial number matches the one pointed to by R1.
	DMKSCNVU	Returns the addresses of the virtual channel, control unit, and device blocks for a given real device to the caller.
DMKSEP		Pageable.
	DMKSEPSP	Prints and punches the respective output separators on real spooling devices.
DMKSEV		Pageable but locked.
	DMKSEV70	Analyzes 2870 channel logout and sets appropriate bits in the ECSW field according to the results of analysis. It moves the channel logout to the check record.
DMKSIX		Pageable but locked. Analyzes 2860 channel logout and sets appropriate bits in the ECSW field according to the results of analysis. It moves the channel logout to the check record.
DMKSNC		Pageable.
	DMKSNCP	Save a page-form version of a 3704/3705 network control program. The name of the network control program and the DASD location at which it is to be saved is defined in the CP module DMKSYS.

Module Name	Entry Points	Attributes, Function
DMKSNT		Pageable.
	DMKSNTBL	This module is assembled by the installation system programmer. It describes the system to be saved via the SAVESYS command and to be initial program loaded by name. Shared segments may be specified. These segments consist of all reenterable code and no altering of this storage is allowed. There is no executable code in this module.
DMKSPL		Pageable. Spool file manager.
	DMKSPLCR	Closes and queues a real reader spool file for virtual input.
	DMKSPLCV	Closes and queues a virtual printer or punch spool file for processing.
	DMKSPLDL	Deletes used files from the system and de-allocates the DASD page space.
	DMKSPLOR	Initializes control blocks and buffers for real input reader files.
	DMKSPLOV	Initializes control blocks and buffers for virtual printer and punch output spool files.
DMKSSP		This module is found in the starter system only. It builds RCHBLOKs, RCUBLOKs, and RDEVBLOKs necessary to configure a minimum CP system. From the starter system, a real CP system figured based on the REALIO deck of the installation.
	DMKSSP01	Entered as a result of an IPL operation. Constructs the I/O blocks and system modules for a minimum system configuration.
DMKSSS		Resident. Services routines for all other modules that require access to the MSS.
	DMKSSSL1	Processes a DEDICATE statement with the 3330V parameter.
	DMKSSSL2	Processes a DEDICATE statement with raddr and volid specified, and the raddr is a 3330V.
	DMKSSSL3	Processes a DEDICATE statement with a volid but no raddr.
	DMKSSSHV	Processes DIAGNOSE code X'78'.
	DMKSSSI1	Reschedules an I/O operation that had previously caused a cylinder fault.
	DMKSSSI2	Queues an I/O request that has just caused a cylinder fault. sets the missing attention handler timer interruption value.
	DMKSSSUS	Quiesces all MSS mount and demount activity.
	DMKSSSVA	Attaches a 3330V to the system or to a virtual machine.
	DMKSSSLN	Allocates a 3330V device and mounts the required 3330V system volume.
	DMKSSSCF	Resets a virtual device defined on a 3330V, including purging any I/O waiting for an MSS volume mount.
	DMKSSSDE	Demounts an MSS volume from a 3330V.

Module Name	Entry Points	Attributes, Function
	DMKSSSEN	Returns to the appropriate requesting routine after an MSS volume mount is complete.
	DMKSSSMQ	Serves as the anchor for the MSSCOM control blocks that are queued for MSS mounts, demounts, and pack change interruptions. Does not contain executable code.
DMKSTK		Resident.
		Stacks I/O blocks.
	DMKSTKCP	Stacks a CPEXBLOK.
	DMKSTKDE	Stacks a deferred execution block.
	DMKSTKIO	Stacks an IOBLOK.
	DMKSTKLF	Stacks a CPEXBLOK LIFO (used by EXTEND and machine check).
	DMKSTKMP	Stacks CPEXBLOK for current processor only.
	DMKSTKOP	Stacks CPEXBLOK for the other processor only.
DMKSVC		Resident.
	DMKSVCIN	Handles any SVC interrupt.
DMKSYM		Pageable.
	DMKSYM	Provides a symbol table of all CSECTS and entry points.
DMKSYS		Resident.
	DMKSYS	Exists as a CSECT that defines the system residence volume, paging space, operator ID, dump ID, storage size, and time zone.
DMKTAP		Pageable.
	DMKTAP	Examines the error condition resulting from a unit check while executing a CP generated tape channel program. Positioning of the tape is required on read/write commands and the channel program is reexecuted. If the error condition is uncorrectable, a call is issued to the message writer (DMKMSW) to notify the operator. Upon regaining control from DMKMSW, the original channel program may be reexecuted or terminated.
	DMKTAPER	Retries the failing tape channel program, after a tape positioning command has been executed.
	DMKTAPRL	Performs tape release to determine two- or four-channel switch capability.
DMKTBL		Resident.
	DMKTBL	Contains the terminal translate tables.
DMKTBM		Pageable.
		Contains terminal translate tables for APL.
	DMKTBMMO	EBCDIC to APL correspondence terminal code. APL correspondence terminal code to APL.
	DMKTBMNI	APL PTTC/EBCD terminal code to EBCDIC.
	DMKTBMNO	EBCDIC to APL PTTC/EBCD terminal code.
	DMKTBMZI	3270 APL compound read translation.
	DMKTBMZO	3270 APL compound write translation.

Module Name	Entry Points	Attributes, Function
DMKTCS		Pageable.
	DMKTCSET	Sets up the 3800 prior to printing the file.
	DMKTCSSP	Sets up the 3800 prior to printing the separator.
	DMKTCSCO	Sets up the forms overlay sequence control.
DMKTDK		Pageable.
	DMKTDKGT	Allocates cylinders of temporary disk space from owned volumes.
	DMKTDKRL	Releases temporary disk space to the pool of free space.
DMKTHI		Pageable. Displays data about use of and contention for major system resources.
	DMKTHIEN	Processes INDICATE command.
DMKTMR		Resident. Simulates the CPU timer and time-of-day clock comparator instructions for virtual machines operating in EC mode.
	DMKTMRCC	Entered after expanded virtual machine assist processing of a virtual SCKC instruction.
	DMKTMRCK	Simulates virtual clock comparator interruptions.
	DMKTMRPT	Calculates user's total virtual problem time.
	DMKTMRSP	Entered after expanded virtual machine assist processing of a virtual SPT instruction.
	DMKTMRTN	Simulates timer instruction.
	DMKTMRVT	Simulates virtual CPU timer interruptions.
DMKTRA		Pageable. Processes the TRACE command line. Provides a virtual machine with facility to track SVC instructions, program interrupts, external interrupts, successful searches, or all instructions with output on the printer or terminal.
	DMKTRACE	TRACE command processor.
DMKTRC		Pageable. Processes the TRACE command functions.
	DMKTRCEX	Traces external interruptions.
	DMKTRCIO	Traces I/O interruptions.
	DMKTRCIT	Sets the needed SVC B2 for instruction tracing.
	DMKTRCND	Ends tracing.
	DMKTRCPB	Puts back user instructions altered by tracing.
	DMKTRCPG	Traces program interruptions.
	DMKTRCPV	Traces privileged instruction interruptions.
	DMKTRCSV	Processes an SVC, Branch, or full instruction TRACE.
	DMKTRCSW	Traces virtual and real CSWs.
DMKTRD		Pageable. Split from DMKTRC.
	DMKTRDSI	Traces I/O operations (SIO, TIO, HIO, TCH).
	DMKTRDWT	Serialization entry for I/O and CCW tracing.

Module Name	Entry Points	Attributes, Function
DMKTRM		Pageable.
	DMKTRMID	Identifies a 2741 terminal as either a 2741P (PTTC/EBCD) or 2741C (correspondence) from the user command. It sets ADEVTYPE the RDEVBLOK to TYP2741P or TYP2741C and sets flag RDEVIDNT on if the terminal was successfully identified.
DMKUCB		Pageable.
	DMKUCB	Contains the UCB buffer load images used by the LOAD command to load the universal character set buffer in the 3811 control unit. This module contains no executable code.
DMKUCC		Pageable.
	DMKUCCLD	Contains the UCB buffer load images used by the LOAD command to load the universal character set buffer in the 3203 printer control unit. This module does not contain executable code.
DMKUCS		Pageable. Contains the UCS buffer load images that the LOAD command uses to load the universal character set buffer in the 2821 control unit. This module does not contain executable code.
DMKUDR		Pageable.
	DMKUDRBV	Allows the DMKDIRCT or DMKCPINT programs to build a list of virtual page buffers; one for each UDIRBLOK page on disk.
	DMKUDRDS	Allows the DMKDIRCT program to swap the active user directory to the newly created user directory.
	DMKUDRFD	Puts specified UDEVBLOK into the caller's buffer.
	DMKUDRFU	Finds a given user ID in the user directory and moves the user's directory entry into the caller's buffer.
	DMKUDRRD	Reads the next user directory into the caller's buffer.
	DMKUDRRV	Releases a virtual page used by the directory program as a buffer.
DMKUDU		Pageable.
	DMKUDUMN	Updates in-place the CP directory on the object DASD page and updates in-place the virtual system page (if used) on the paging device. Entered from DMKHVD when a class B virtual machine issues a DIAGNOSE code '84' instruction.
DMKUNT		Resident. Untranslates CCWs and CSWs.
	DMKUNTFR	Releases pages and free storage used for the CCW chain. Also processes the CP assist instruction, UNTFR (E605).
	DMKUNTIS	Finds the RCWTASKS that have been patched to handle OS ISAM self-modifying sequences and put them back the way DMKCCW had them to allow DMKUNTRN and DMKUNTFR to operate correctly.

Module Name	Entry Points	Attributes, Function
	DMKUNTRN	Translates a real CSW into a virtual CSW. Also processes the CP assist instruction, UNTRN (E610).
	DMKUNTRS	Relocates sense byte information. For a 3330, 3340, 3350, or 2305, computes virtual cylinder member in bytes 5 and 6 of the sense byte data by unrelocating the real cylinder number given by the hardware. For a 2311 simulated on a 2314 or 2319, computes the appropriate status for byte 3 of the sense data from the real sense data given by the hardware.
DMKUSO		Pageable. Processes user termination.
	DMKUSODS	Processes the DISCONN (disconnect) command.
	DMKUSOFF	Logs off a user.
	DMKUSOFL	Processes the FORCE command.
	DMKUSOFM	Returns subpools from the free storage chain and removes spool file blocks and allocation blocks from the dynamic paging area.
	DMKUSOLG	Processes the LOGOFF command.
DMKVAT		Resident. Storage management for EC mode virtual machine.
	DMKVATAB	Allocates, initializes and maintains shadow, segment, and page tables for virtual machines that can relocate.
	DMKVATBC	Returns active shadow tables to free storage.
	DMKVATEX	Services page or segment exceptions for virtual EC machines.
	DMKVATLA	Virtual - virtual to virtual address translation.
	DMKVATMD	Allocates and initializes shadow tables.
	DMKVATPF	Handles pseudo page fault interruption from a VS1 virtual machine.
	DMKVATPX	Processes paging exceptions for a virtual machine that performs paging.
	DMKVATRN	Virtual (shadow) -- virtual-to-real address translation.
	DMKVATSX	Processes segment exception for a virtual machine that performs paging.
	DMKVATZP	Processes the CP assist instruction, ZAPPAGE (E60B).
	DMKVATZS	Processes the CP assist instruction, ZAPSEGS (E60A).
DMKVCA		Pageable. Simulates I/O for a virtual channel-to-channel adapter.
	DMKVCARD	Selectively resets a device for a virtual channel-to-channel adapter without decoupling the CTCA from the Y-side adapter.
	DMKVCARS	Does a final reset for a virtual channel-to-channel adapter and and disconnects the adapter from its coupled twin on the Y-side virtual machine.

Module Name	Entry Points	Attributes, Function
	DMKVCASH	Simulates the execution of a HALT I/O or HALT DEVICE instruction for a virtual machine channel-to-channel adapter.
	DMKVCAST	Simulates the channel and device operations of the channel-to-channel adapter (CTCA) connected between two virtual machines under VM/370.
	DMKVCATS	Simulates the TEST I/O instruction for a virtual channel-to-channel adapter that has no interruptions pending.
DMKVCH		Pageable.
	DMKVCHDC	Processes the ATTACH and DETACH real devices and channels) command.
DMKVCN		Resident.
	DMKVCNEX	Simulates all SIOs to a virtual console.
DMKVDA		Pageable.
	DMKVDAAT	Handles the ATTACH command. Attaches a real device to a user as a virtual device, or dedicates all devices on a particular channel to a specified user.
DMKVDC		Pageable.
	DMKVDCAL	Chains RDEVBLOKs off the allocation chain anchors.
	DMKVDCPS	Acquires virtual blocks for devices that are likely to be attached by the ATTACH command.
	DMKVDCSC	Scans the ATTACH and DETACH command lines and checks syntax.
DMKVDD		Pageable.
	DMKVDDDE	Handles the DETACH command.
DMKVDE		Pageable.
	DMKVDEDC	Verifies the existence of a device specified on an ATTACH command.
DMKVDR		Pageable.
	DMKVDREL	Releases a virtual or real device from a virtual user.
DMKVDS		Pageable.
	DMKVDSAT	Attaches a virtual device to a user.
	DMKVDSDF	Defines a new virtual device for user.
	DMKVDSLK	Links a virtual DASD device to a user.
DMKVER		Pageable.
		Processes error records from virtual machine via SVC 76.
	DMKVERD	Processes SVC 76 from DOS or DOS/VS.
	DMKVERO	Processes SVC 76 from OS, VS/1, VS/2, or VM/370.

Module Name	Entry Points	Attributes, Function
DMKVIO		Resident. Records and translates the interrupts and status associated with virtual I/O operations.
	DMKVIOC1	Reflects condition code 1 CSW status.
	DMKVIOIN	Translate a virtual I/O interruption.
	DMKVIOMK	Address of a table of interruption masks, indexable by device address.
DMKVMA		Resident.
	DMKVMASH	Checks all protected shared pages associated with shared named systems and determines if they have been changed. If they were changed, the page is returned to CP free storage and the condition code is made nonzero.
	DMKVMASW	Switches the user's segment table entries from one protected shared page table to the other.
DMKVMC		Pageable.
	DMKVMCFC	Main entry for all VMCF subfunctions. Called by DMKHVC when a DIAGNOSE X'0068' instruction is executed. Builds a VMCBLOK with information from user-supplied parameter list, validates the subfunction code, and passes control to appropriate VMCF subroutine.
	DMKVMCEX	Called by DMKDSP to reflect the VMCF external interrupt message header and optional SENDX data to a virtual machine. Copies the message header from the VMCBLOK to the user's external interrupt buffer. If interrupt is for a SENDX request, move SENDX data to the optional area in the external interrupt buffer.
	DMKVMCVA	Branched to from the DMKVMCFC entry point or called by DMKCFP during a system reset. Releases the master VMCBLOK and any final response VMCBLOKs (VMCRESP bit). Returns other VMCBLOKs to the original SOURCE users with the notification that this user is not available.
DMKVMI		Pageable. Loaded into the user's virtual storage when invoked. Performs an IPL of a virtual machine.
	DMKVMIPL	Simulates a user's IPL sequence.
DMKVSI		Resident. Simulates the operation of privileged I/O instructions issued by virtual machines.
	DMKVSIEX	Simulates a SIO, TIO, HIO, TCH, or CLCH.
	DMKVSISC	Scans a V=R channel program for exceptional conditions, such as sense commands, no-ops, I/O to and from page 0, etc., without actually translating the program.
DMKVSP		Resident. Simulates all user SIOs to a virtual unit record device (real reader, punch, print, or pseudo timer that is spooled rather than dedicated. It also handles control program requests to print on the user's virtual printer.

Module Name	Entry Points	Attributes, Function
	DMKVSPCO	Stops processing the file currently in the spooled printer or punch and clears all pending status from the spooled printer or punch.
	DMKVSPCP	Writes a print line to the console.
	DMKVSPCR	Stops processing the file currently in the spooled card reader and clears all pending status from the spooled card reader.
	DMKVSPEX	Simulates SIO to a spooled unit record device.
	DMKVSPRT	Puts a CP-generated line on the user's spooled printer.
	DMKVSPTO	Checks if the virtual reader is empty.
	DMKVSPVP	Simulates SIO to a spooled virtual console.
	DMKVSPWA	Nonexecutable index work area for 2311.
DMKWRM		Pageable.
	DMKWRMST	Warm start processing. Retrieves the system log messages, accounting cards, spool file blocks, and spooling allocation records from the warm start cylinder on the IPL pack.

MODULE	EXTERNAL REFERENCES (LABELS AND MODULES)												
DMKACO	ACCTBLOK	ACCTUSER	ACNTBACK	ACNTBLOK	ACNTCCW	ACNTCODE	ACNTCONT	ACNTDATA	ACNTDEVC	ACNTIOCT	ACNTNCYL	ACNTNEXT	ACNTNUM
	ACNTPGRD	ACNTSIZE	ACNTSTOP	ACNTTIME	ACNTUSER	ACNTVTIM	ACOACCL	ACOACCM	ACOCHK	ACOEXIT	ACORETBL	ACORETN	ADSPCH
	AEXTSP	AFREE	AFRET	ALARM	APSTAT1	APTRLK	APUOPER	AQCNWT	ARIODV	ARIOPU	ARSPAC	ASYSLC	ASYSVM
	CC	CLASDASD	CLASGRAF	CLASTERM	CORCP	CORFLAG	CORTABLE	CPEXADD	CPEXBLOK	CPEXSIZE	DATE	DE	DEVCARD
	DFRET	DISPMSG	DMKCVTAB	DMKCVTBH	DMKDSPCH	DMKERMSG	DMKFREE	DMKFRET	DMKIOSQR	DMKLOKSW	DMKPTRLK	DMKPTRUL	DMKQCNRD
	DMKQCNWT	DMKRSPEX	DMKSCHDL	DMKSTKCP	DMKSTKIO	DMKSYSCK	DMKTMRPT	FTROPRDR	F1	F4	F4095	F60	F8
	INHIBIT	IOBCAW	IOBCP	IOBCSW	IOBFATAL	IOBFLAG	IOBIRA	IOBLINK	IOBLOK	IOBMISC	IOBMISC2	IOBRADD	IOBSIZE
	IOBSPEC	IOBSTAT	IOBUSER	IPUADDRX	LOCGRAF	LOCK	NICBLOK	NICOPRDR	NICSIZE	NICTYPE	NORET	PRIORITY	PROCIO
	PSA	RDEVACNT	RDEVBLOK	RDEVBUSY	RDEVCLAS	RDEVDED	RDEVDISA	RDEVDRAN	RDEVFLAG	RDEVFTR	RDEVMDL	RDEVNICL	RDEVSPL
	RDEVSTAT	RDEVTMAT	RDEVTYPC	RDEVTYPE	R0	R1	R10	R11	R12	R13	R14	R15	R2
	R3	R4	R5	R6	R7	R8	R9	SAVEAREA	SAVEREGS	SAVER11	SAVER2	SAVEWRK1	SAVEWRK2
	SAVEWRK3	SAVEWRK6	SAVEWRK7	SAVEWRK8	SAVEWRK9	SETUP	SETUP1	SETUP2	SIGEMS	SIGQUI	SIGRES	SIGXC	SILI
	SKIP	START	STCODE	STOP	SYSLOCS	TIMEDISP	TODATE	TYPBSC	TYP2540P	TYP3277	USERCARD	VDEVBLOK	VDEVBND
	VDEVFLAG	VDEVREAL	VDEVTDSK	VDEVTMAT	VDEVTYPC	VMACNT	VMACOUNT	VMAPTIME	VMBLOK	VMCFWAIT	VMCPTIME	VMCRDS	VMDSTAT
	VMEXWAIT	VMIDLE	VMINQ	VMIOCNT	VMLINS	VMLOGOFF	VMLONGWT	VMNORUN	VMPGREAD	VMPGWRIT	VMPNCH	VMRSTAT	VMTERM
	VMTIMEON	VMTMINQ	VMTRMID	VMTTIME	VMUSER	VMVTIME	ZEROES						
DMKALG	ADSPCH	AFREE	APSTAT1	APUOPER	BUFCNT	BUFFER	BUFNXT	BUFSIZE	CPEXADD	CPEXBLOK	CPEXR0	CPEXR12	CPEXSIZE
	DMKBLDVM	DMKERMSG	DMKFREE	DMKLOGB	DMKLOKSW	DMKSCHDL	DMKSCNVD	DMKSTKCP	DMKSYSJR	F1	F240	JPSCBLOK	JPSLOGDS
	LOCK	LOGONJRL	PSA	R0	R1	R10	R11	R12	R13	R14	R15	R2	R3
	R4	R7	R8	R9	SAVEAREA	SAVEREGS	SAVER11	SAVER2	SAVER9	SAVEWRK1	SAVEWRK8	TIMEDISP	VCONCTL
	VCONRBSZ	VCONRBUF	VCONRCNT	VDEVAUCR	VDEVBLOK	VDEVCFLG	VDEVCCN	VMBLOK	VMCF	VMCFWAIT	VMCOMND	VMDISC	VMDVSTRT
	VMKILL	VMOSTAT	VMPSWDCA	VMPSWDCT	VMRSTAT	VMSLEEP	VMVIRCF	VMVTERM					
DMKAPI	ACTIVTRQ	AFREE	ALOKVM	APSTAT1	APSTAT4	APTRAN	APTRLK	APUOPER	AQCNWT	ASYSOP	ASYSVM	BALRSAVE	BALR1
	BLKMPX	BRING	CKCMASK	CPCREG0	CPEXSIZE	CPINITD	CPMICAVL	CPMICON	CPSTATUS	CPSTAT2	CPTMASK	CPUID	CPWAIT
	C0	C1	C14	C2	C6	DEFER	DMKFREE	DMKIOGAP	DMKLCKDF	DMKLOKSY	DMKPRGIN	DMKPSADU	DMKPTRAN
	DMKPTRUL	DMKQCNWT	DMKSVCIN	EMSMASK	EXOPSW	EXTMASK	EXTMODE	FFS	F1	F4096	IDLEWAIT	INTMASK	IONTWAIT
	IPUADDR	IPUADDRX	KEYMASK	LASTUSER	LOCK	LPUADDR	LPUADDRX	MCHEK	MFAMASK	PAGEWAIT	PAGE4K	PGREAD	POFFLINE
	PREFIXA	PREFIXB	PRNPSW	PROBTIME	PROCIO	PSA	PSENDCLR	RSRTNPSW	RUNCR0	RUNCR1	RUNUSER	R0	R1
	R10	R11	R12	R13	R14	R15	R2	R3	R4	SAVEAREA	SAVEREGS	SAVER11	SIGREST
	START	SVCNPSW	SYSTEM	TEMPR0	TEMPR2	TEMPR3	TEMPR4	TEMPR5	TEMPSAVE	TIMEDISP	TIMER	TRACPROC	TRACSTRT
	TYPE	VMAPTIME	VMBLOK	VMCPTIME	VMDFTPNT	VMMFE	VMMSVC	VMPNT	VMSEG	VMTTIME	WAIT	XCMASK	ZEROES
DMKATS	ACORETBL	AFREE	AFRET	APSTAT1	APSTAT2	APTRLK	APUOPER	ARIODV	ASYSVM	BRING	CORBPNT	CORCFLCK	CORFLAG
	CORFLUSH	CORFPNT	CORFREE	CORIOLCK	CORPGPNT	CORRSV	CORSHARE	CORSWPNT	CORTABLE	CORVM	CPPTLBR	C1	DEFER
	DMKCVTAB	DMKDSPNP	DMKERMSG	DMKFREE	DMKFRET	DMKPGTPR	DMKPTRAN	DMKPTRFT	DMKPTRPW	DMKPTRRC	DMKPTRSC	DMKPTRUC	DMKPTRUL
	DMKSCNVS	DMKSNTBL	DMKSYSAP	DMKSYSOW	DMKVMASH	F1	F15	F16	F256	F4	F4096	F8	LASTUSER
	LOCK	LPUADDR	MPFEAT	OWNDLIST	OWNDRDEV	PAGACT	PAGBMP	PAGCORE	PAGINVAL	PAGSHR	PAGSTMP	PAGSWP	PAGTABLE
	PAGTOT	PAGTSWP	PREFIXA	PROCIO	PSA	RDEVBLOK	RDEVCCDE	RDEVTYPE	R0	R1	R10	R11	R12
	R13	R14	R15	R2	R3	R4	R5	R6	R7	R8	R9	SAVEAREA	SAVEREGS
	SAVEWRK1	SAVEWRK2	SAVEWRK3	SAVEWRK4	SAVEWRK5	SAVEWRK6	SAVEWRK8	SAVEWRK9	SEGINV	SEGPAGE	SEGTABLE	SHRBPNT	SHRFLAG
	SHRFPNT	SHRNAME	SHRNOPRT	SHRPAGE	SHRSEGCT	SHRSEGNM	SHRTABLE	SHRTSIZE	SHRUSECT	SWPALLOC	SWPAPP	SWPCHG1	SWPCODE
	SWPCYL	SWPFLAG	SWPFLAG2	SWPKEY1	SWPPAG	SWPRECMP	SWPSHR	SWPTABLE	SWPVM	SWPVPAGE	SYSNAME	SYSPAGLN	SYSPAGNM
	SYSPNT	SYSSTART	SYSTBL	SYSTEM	SYSVOL	TEMPR0	TEMPR11	TEMPR2	TTSEGCNT	TYP2305	TYP2314	TYP3330	TYP3350

MODULE	EXTERNAL REFERENCES (LABELS AND MODULES)												
	VMABLOK	VMAFPNT	VMANAME	VMASIZE	VMASSIST	VMBLOK	VMOSTAT	VMPAGES	VMPDISK	VMPDRUM	VMSEG	VMSHR	VMSHRPRC
	VMSHRSYS	XPAGNUM											
DMKBLD	ACORETBL	AFREE	AFRET	APSTAT1	APSTAT2	APTRAN	APUOPER	AQCNWT	ASYSLC	ASYSVM	AVMREAL	CLASGRAF	CLASSPEC
	CLASTERM	CORCFLCK	CORFLAG	CORFPNT	CORIOLCK	CORLCNT	CORPGPNT	CORSWPNT	CORTABLE	CPEXSIZE	CPPTLBR	C1	DEFER
	DELPAGES	DELSEGS	DMKCVTAB	DMKCVTBH	DMKERMSG	DMKFREE	DMKFRET	DMKLOKDF	DMKPTRAN	DMKQCNWT	DMKRIORN	DMKSCHCP	DMKSCNRD
	DMKSYSLE	DMKSYSLL	DMKTMRCK	ECBLOK	EXTCCTRQ	EXTCPTRQ	EXTCR0	EXTCR14	EXTCR15	EXTCR2	EXTSIZE	FFS	F1
	F15	F16	F255	F4	F4095	F7	F8	KEEPSEGS	LASTUSER	LOCK	LOCKSAV	LPUADDR	MICBLOK
	MICRSEG	NEWPAGES	NEWSEGS	NICBLOK	NICCIBM	NICGRAF	NICLLEN	NICNAME	NICTERM	NICTYPE	NICUSER	NORET	OLDVMSEG
	PAGACT	PAGBMP	PAGCORE	PAGSTMP	PAGSWP	PAGTABLE	PAGTONLY	PREFIXA	PREFIXB	PSA	RDEVBLOK	RDEVFLAG	RDEVLLEN
	RDEVPSUP	RDEVTYPC	RDEVTYPE	RDEVUSER	R0	R1	R10	R11	R12	R13	R14	R15	R2
	R3	R4	R5	R6	R7	R8	R9	SAVE	SAVEAREA	SAVEREGS	SAVER1	SAVER11	SAVER2
	SAVER8	SAVEWRK1	SAVEWRK2	SAVEWRK9	SEGENQ	SEGINV	SEGPAGE	SEGPLEN	SEGTABLE	START	STARTIME	STOP	SWPFLAG
	SWPPAG	SWPRECMP	SWPTABLE	SWPVM	SYSLOCS	TIMEDISP	TRQBIRA	TRQBLOK	TRQBSIZE	TRQBUSER	TTSEGCNT	TYPBSC	TYPE
	TYP3705	VMAEX	VMAPTIME	VMBLOK	VMBSIZE	VMCFWAIT	VMCHTBL	VMCONBUF	VMCPTIME	VMDFTPNT	VMECEXT	VMEPRIOR	VMESTAT
	VMGRFTAB	VMINVPAG	VMLOCK	VMLOCKER	VMLOGON	VMMCODE	VMMICRO	VMMLEVEL	VMMLINED	VMMSGON	VMMTEXT	VMPAGES	VMPNT
	VMPSTAT	VMPSW	VMQLEVEL	VMREAL	VMRSTAT	VMSEG	VMSEGDSP	VMSIZE	VMSTOR	VMTERM	VMTLEND	VMTMOUTQ	VMTODINQ
	VMTRMID	VMTTIME	VMUSER	VMVTERM	VMV370R	VMWNGON	VMWSPROJ	VRALOC	WAIT	ZEROES			
DMKBSC	AFREE	AFRET	APSTAT1	BSCBLOK	BSCPCCW1	BSCPCCW2	BSCREAD	BSCRESP	CC	CCC	CDC	CHC	DMKFREE
	DMKFRET	DMKIOEST	DMKMSWR	FAILCCW	FTRDIAL	F1	F15	F7	F8	IFCC	IOBCAW	IOBCSW	IOBERP
	IOBFATAL	IOBFLAG	IOBIOER	IOBLOK	IOBRCAW	IOBRCNT	IOBRSTRT	IOBSTAT	IOERACT	IOERBLOK	IOERCAN	IOERCCRA	IOERCCRL
	IOERCSW	IOERDATA	IOERDW	IOEREXT	IOERFLG2	IOERFLG3	IOERIND3	IOERINFO	IOERLOC	IOERMSW	IOERNUM	IOERREAD	IOERSIZE
	LOCK	PRGC	PROCIO	PRTC	PSA	RDEVBLOK	RDEVBSC	RDEVFTR	RDEVIOER	R0	R1	R10	R11
	R12	R13	R14	R15	R2	R3	R4	R5	R6	R7	R8	R9	SAVEAREA
	SAVEREGS	SILI	UC	WRITE	WRITEOT	WRITE1	ZEROES						
DMKCCH	AFREE	AFRET	ALARM	AMCHAREA	APTRAN	ARIOCH	ARIOCT	ARIOCU	ARIODV	BRING	CAW	CCC	CCCPUID
	CCDEVTYP	CCHADDR	CCHANID	CCHCAV	CCHCLOGL	CCHCUA	CCHHIC	CCHINTB	CCHIOH	CCHLOG45	CCHLOG80	CCHRCV	CCHREC
	CCHSIOB	CCHSIZE	CCHSIZE1	CCHSNSB	CCHTIO	CCPROGID	CCRECTYP	CDC	COMPSYS	CPUID	CSW	C1	C7
	DEFER	DEVCCH	DMKCVTBH	DMKFREE	DMKFRET	DMKIOECC	DMKMCHST	DMKPTRAN	DMKQCNWT	DMKSCNRU	DMKSYSRM	ECSWBYT3	ECSWLOG
	FAILADD	FAILCCW	FAILCSW	FAILECSW	FFS	FXDLOG	F1	F16	F255	F7	F8	HIOCCH	IFCC
	IGPRGFLG	IGTERMSQ	IGVALIDB	INTERCCH	INTTIO	IOBCCH	IOBCP	IOBCSW	IOBFLAG	IOBHIO	IOBIOER	IOBLOK	IOBRADD
	IOBSPEC	IOBTIO	IOBUSER	IOELPNTR	IOERBLOK	IOERB80	IOERCCH	IOERCCRA	IOERCCRL	IOERCCUA	IOERCHID	IOERCLOG	IOERCSW
	IOERECSW	IOEREXT	IOERLG45	IOERLOGL	IOERSIZE	IOERS80	IOERZCSW	IOER2860	IOER2870	IOOPSW	LOCK	MCHAREA	MCHMODEL
	MCNPSW	MODEL135	MODEL145	MODEL155	MODEL165	MOD4331	NORET	OPERATOR	PSA	RCHADD	RCHBLOK	RCHCUTBL	RCHDED
	RCHSTAT	RCHSTIDC	RCUADD	RCUBLOK	RCUDVTBL	RDEVADD	RDEVAIOB	RDEVBLOK	RDEVBUSY	RDEVSTAT	RESTDEV	R0	R1
	R10	R11	R12	R13	R14	R15	R2	R3	R4	R5	R6	R7	R8
	R9	SAVEAREA	SAVEREGS	SAVER4	SAVEWRK1	SAVEWRK2	SAVEWRK3	SAVEWRK4	SAVEWRK5	SAVEWRK6	SAVEWRK7	SAVEWRK8	SAVEWRK9
	SIOCCH	TERMSYS	TIOCCH	VDEVBLOK	VDEVIOER	VMBLOK	VMDVSTRT	VMIOLOG	VMSEG	VMUSER	VMVCR14	ZEROES	
DMKCCW	ACORETBL	ADSPCH	AFREE	AFRET	APTRAN	APTRLK	BALRSAVE	BALR2	BALR3	BRING	CC	CD	CLASDASD
	CLASGRAF	CLASSPEC	CLASTAPE	CLASTERM	CLASURI	CLASURO	CORFLAG	CORFLUSH	CORSHARE	CORSWPNT	CORTABLE	CPEXADD	CPEXBLOK
	CPEXFPNT	CPEXMISC	CPEXR0	CPEXSIZE	CPSHRLK	CPSTAT2	C1	DEFER	DMKDIBSM	DMKDSESD	DMKDSPCH	DMKFREE	DMKFRET
	DMKISMTR	DMKPTRAN	DMKPTRFR	DMKPTRLK	DMKPTRUL	DMKRSESD	DMKSCNRU	DMKSCNVD	DMKSYSCS	DMKSYSRM	DMKTRKVA	DMKUNTFR	DMKUNTRS
	DMKVMASH	FFS	FTREXTSN	FTR35MB	F1	F10	F15	F16	F2	F240	F3	F4	F4095

MODULE	EXTERNAL REFERENCES (LABELS AND MODULES)												
	F4096	F7	F8	F9	IDA	IOBALTSK	IOBCAW	IOBCLN	IOBCYL	IOBFLAG	IOBLOK	IOBMISC	IOBMISC2
	IOBRELCU	IOBSIZE	IOBSPEC2	IOBSTAT	IOBUNREL	IOBWRAP	IOERBLOK	IOERDATA	IOEREXT	IOERLEN	IOERSIZE	LOCK	NOP
	PCIF	PSA	RCUBLOK	RCUCHB	RCUPRIME	RCUSUB	RCUTYPE	RCWADDR	RCWCCNT	RCWCCW	RCWCNT	RCWCOMND	RCWCTL
	RCWFLAG	RCWGEN	RCWHEAD	RCWHMR	RCWINVL	RCWIO	RCWISAM	RCWPNT	RCWRCNT	RCWREL	RCWSHR	RCWTASK	RCWVCAW
	RCWVCNT	RCW2311	RDEVBASE	RDEVBLOK	RDEVCUA	RDEVCUB	RDEVFTR	RDEVSADN	RDEVTYPC	RDEVTYPE	RDEVUSER	R0	R1
	R10	R11	R12	R13	R14	R15	R2	R3	R4	R5	R6	R7	R8
	R9	SAVEAREA	SAVEREGS	SAVER1	SAVER10	SAVER12	SAVER2	SAVER8	SAVER9	SAVEWRK1	SAVEWRK2	SAVEWRK4	SAVEWRK9
	SHRLKCNT	SILI	SKIP	SWPFLAG	SYSVIRT	TAPE	TEMPR10	TEMPR14	TEMPR15	TEMPR2	TEMPR3	TEMPSAVE	TYPUNSUP
	TYP1442R	TYP2305	TYP2311	TYP2314	TYP2955	TYP3210	TYP3277	TYP3278	TYP3330	TYP3340	TYP3350	TYP3410	TYP3420
	TYP3704	TYP3705	TYP3800	TYP3851	VIEVBLOK	VDEVBND	VDEVCPEX	VDEVDED	VDEVDIAL	VDEVENAB	VDEVFLAG	VDEVFLG2	VDEVIOER
	VDEVPOSN	VDEVRDO	VDEVREAL	VDEVRELN	VDEVRES	VDEVRRB	VDEVRRF	VDEVRSRL	VDEVSAS	VDEVSTAT	VDEVTYPC	VDEVTYPE	VDEVUC
	VDEV231B	VDEV231T	VIRTUAL	VMBLOK	VMCLASSF	VMCLEVEL	VMDVSTRT	VMFAUTO	VMFSTAT	VMIDLE	VMISAM	VMOSTAT	VMPSTAT
	VMRSTAT	VMSEG	VMSHR	VMSIZE	VRRADD	VRRBLOK	VRRCPEX	VRRRES	VRRSTAT	VRRUSER	XPAGNUM	XRIGHT16	XRIGHT24
	X2048BND	ZEROES											
DMKCDB	AFREE	AFRET	APSTAT1	APTRAN	APUOPER	AQCNWT	BRING	C1	DEFER	DMKCVTBD	DMKCVTBH	DMKCVTDB	DMKCVTFP
	DMKCVTHB	DMKDMPTR	DMKERMSG	DMKFREE	DMKFRET	DMKPTRAN	DMKQCNWT	DMKSCNFD	DMKSYSAP	DMKSYSRM	DMKVATAB	ECBLOK	ERRPARM
	EXTCR0	FFS	F1	F10	F15	F16	F2	F24	F3	F4	F4095	F4096	F6
	INVLD	LOCK	NORET	PREFIXA	PREFIXB	PROCIO	PSA	RANGE	R0	R1	R10	R11	R12
	R13	R14	R15	R2	R3	R4	R5	R6	R7	R8	SAVEAREA	SAVEREGS	SAVEWRK1
	SAVEWRK4	SAVEWRK5	SAVEWRK6	SAVEWRK8	VMBLOK	VMECEXT	VMESTAT	VMEXTCM	VMFPRS	VMGPRS	VMINVPAG	VMINVSEG	VMKILL
	VMLOGOFF	VMNEWCR0	VMOSTAT	VMPSTAT	VMPSW	VMRSTAT	VMSEG	VMSHR	VMSIZE	VMVCR0	VMV370R	XPAGNUM	X40FFS
	ZEROES												
DMKCDM	AFREE	AFRET	APSTAT1	APTRAN	APUOPER	AQCNWT	BRING	BUFFER	BUFNXT	C1	DEFER	DMKCVTBD	DMKCVTBH
	DMKCVTDB	DMKCVTFP	DMKCVTHB	DMKDMPTR	DMKERMSG	DMKFREE	DMKFRET	DMKPTRAN	DMKQCNWT	DMKSCNFD	DMKSYSAP	DMKSYSRM	DMKVATAB
	DMKVMASH	DMKVSPRT	ECBLOK	ERRPARM	EXTCR0	FFS	F1	F10	F2	F24	F3	F4	F4096
	F6	INVLD	LASTUSER	LOCK	NORET	PREFIXA	PREFIXB	PROCIO	PSA	RANGE	R0	R1	R10
	R11	R12	R13	R14	R15	R2	R3	R4	R5	R6	R7	R8	R9
	SAVEAREA	SAVEREGS	SAVEWRK1	SAVEWRK2	SAVEWRK4	SAVEWRK5	SAVEWRK6	SAVEWRK8	VMBLOK	VMECEXT	VMESTAT	VMEXTCM	VMFPRS
	VMGPRS	VMINVPAG	VMINVSEG	VMKILL	VMLOGOFF	VMNEWCR0	VMOSTAT	VMPSTAT	VMPSW	VMRSTAT	VMSEG	VMSHR	VMSIZE
	VMVCR0	VMV370R	XPAGNUM	XRIGHT16	X40FFS	ZEROES							
DMKCDS	ACORETBL	AFREE	AFRET	APSTAT1	APTRAN	APUOPER	AQCNWT	BLANKS	BRING	CORFLAG	CORPGPNT	CORSHARE	CORSWPNT
	CORTABLE	CPEXADD	CPEXBLOK	CPEXFPNT	CPEXR0	CPEXR14	CPEXR15	CPEXR5	CPEXR7	CPEXSIZE	C1	DEFER	DMKATSCF
	DMKCVTBH	DMKCVTDB	DMKCVTHB	DMKERMSG	DMKFREE	DMKFRET	DMKPAGIO	DMKPGTPG	DMKPSACC	DMKPSASC	DMKPTRAN	DMKPTRWQ	DMKQCNWT
	DMKSCNFD	DMKSYSAP	DMKSYSRM	DMKTRCIT	DMKTRCPB	DMKVATAB	DMKVATBC	DMKVATMD	DMKVMASH	ECBLOK	EXTCCTRQ	EXTCPTMR	EXTCR0
	EXTMODE	F1	F15	F16	F2	F4	F5	F6	F8	LOCK	NORET	PAGCORE	PAGINVAL
	PREFIXA	PREFIXB	PROCIO	PSA	RUNUSER	R0	R1	R10	R11	R12	R13	R14	R15
	R2	R3	R4	R5	R6	R7	R8	R9	SAVEAREA	SAVEREGS	SAVER2	SAVEWRK1	SAVEWRK2
	SAVEWRK3	SAVEWRK4	SAVEWRK5	SAVEWRK8	SHRPAGE	SWPCYL	SWPFLAG	SWPRECMP	SWPTRANS	TEMPR14	TEMPR15	TRANMODE	TRQBLOK
	TRQBVAL	VMBLOK	VMECEXT	VMESTAT	VMEXTCM	VMFPRS	VMGPRS	VMINVPAG	VMINVSEG	VMNEWCR0	VMPSTAT	VMPSW	VMSEG
	VMSIZE	VMTIMER	VMTRBRIN	VMTRCTL	VMUSER	VMVCR0	VMV370R	WAIT	XPAGNUM	ZEROES			
DMKCFC	AFREE	AQCNWT	ATTN	BLANKS	DMKALGON	DMKCDBDC	DMKCDEDI	DMKCDMDM	DMKCDMDU	DMKCDSCP	DMKCDSTO	DMKCFDAD	DMKCFDLO
	DMKCFG1P	DMKCFHSV	DMKCFMAT	DMKCFMWU	DMKCFOEX	DMKCFSET	DMKCFTRM	DMKCPBEX	DMKCPBNR	DMKCPBRS	DMKCPBRW	DMKCPBRY	DMKCPBSR

MODULE	EXTERNAL REFERENCES (LABELS AND MODULES)												
	DMKCPSH	DMKCPSRY	DMKCPSSH	DMKCPVAC	DMKCPVDS	DMKCPVEN	DMKCPVLK	DMKCPVUL	DMKCQGEN	DMKCQPRV	DMKCQREY	DMKCQYEY	DMKCSBLD
	DMKCSBVL	DMKCSOBS	DMKCSODR	DMKCSOFL	DMKCSORP	DMKCSOSP	DMKCSOST	DMKCSPSP	DMKCSQCL	DMKCSQFR	DMKCSQHL	DMKCSTAG	DMKCSUCH
	DMKCSVOR	DMKCSVPU	DMKCSVTR	DMKCVTAB	DMKCVTDB	DMKCVTHB	DMKDEFIN	DMKDIACP	DMKDIAL	DMKERMSG	DMKFREE	DMKJRLQU	DMKJRLSE
	DMKLNKIN	DMKLOGON	DMKMCCCL	DMKMSGEC	DMKMSGMS	DMKMSGNH	DMKMSGSM	DMKMSGWN	DMKNETWK	DMKQCNWT	DMKSCHRT	DMKSCHST	DMKSCNFD
	DMKTHIEN	DMKTRACE	DMKTRCIT	DMKTRCPB	DMKUSODS	DMKUSOFL	DMKUSCLG	DMKVDAAT	DMKVIDDE	FFS	F1	F2	F3
	F4	F6	F60	F8	HOLD	LOCK	NORET	PSA	RDEVBLOK	RDEVTYPE	RESET	RUN	R0
	R1	R11	R12	R13	R14	R15	R2	R3	R4	R5	R6	R7	R8
	R9	SAVEAREA	SAVEREGS	SAVERETN	SAVER1	SAVER2	SAVEWRK1	SAVEWRK2	SAVEWRK4	SEARCH	START	SYSTEM	TRQBIRA
	TRQBLOK	TRQBSIZE	TRQBTOD	TRQBUSER	TRQBVAL	TYPE	TYP2741	VMBLOK	VMCLASSA	VMCLASSB	VMCLASSC	VMCLASSD	VMCLASSE
	VMCLASSF	VMCLASSG	VMCLASSH	VMCLEVEL	VMCOMND	VMDELAY	VMLOGCN	VMOSTAT	VMPSW	VMRSTAT	VMSLEEP	VMTERM	VMTRBRIN
	VMTRCTL	VMVIRCF											
DMKCFD	AFREE	AFRET	APTRAN	AQCNWT	BLANKS	BRING	C1	DEFER	DMKATSCF	DMKCVTBH	DMKCVTHB	DMKERMSG	DMKFREE
	DMKFRET	DMKPSASC	DMKPTRAN	DMKQCNWT	DMKSCNAU	DMKSCNFD	DMKSCNRU	DMKSCNVU	F1	F3	F6	NORET	PSA
	R0	R1	R10	R11	R12	R13	R14	R15	R2	R3	R4	R5	R6
	R7	R8	SAVEAREA	SAVEREGS	SAVEWRK1	SAVEWRK2	SAVEWRK3	SAVEWRK4	SAVEWRK5	SAVEWRK7	VMADSTOP	VMBLOK	VMESTAT
	VMMCR6	VMMICSVC	VMMSVC	VMSEG	VMSHRSYS	VMSIZE	ZEROES						
DMKCFG	AFREE	AFRET	APSTAT1	APTRAN	APTRLK	APUOPER	ASYSVM	AVMREAL	BRING	BUFFER	BUFNXT	C1	DEFER
	DMKBLDRT	DMKCFPRR	DMKCVTBH	DMKCVTDB	DMKCVTHB	DMKERMSG	DMKFREE	DMKFRET	DMKPGSPO	DMKPGSPP	DMKPGSPR	DMKPGSPS	DMKPTRAN
	DMKPTRUL	DMKRPAGT	DMKSCNFD	DMKSCNVS	DMKSCNVU	DMKSNTBL	DMKSYSAP	DMKVATMD	DMKVMAS1	DMKVMAS2	DMKVMI	ECBLOK	EXTCR0
	EXTMODE	F0	F1	F15	F16	F2	F256	F3	F4	F4095	F4096	F7	F8
	KEEPSEGS	LOCK	LPUADDR	MPFEAT	NEWPAGES	OLDVMSEG	PAGACT	PAGBMP	PAGCORE	PAGSHR	PAGSWP	PAGTABLE	PAGTOT
	PAGTSWP	PROCIO	PSA	RDEVBLOK	RDEVCODE	RDEVFLAG	RDEVOWN	RDEVSER	RDEVTYPE	R0	R1	R10	R11
	R12	R13	R14	R15	R2	R3	R4	R5	R6	R7	R8	R9	SAVCREGS
	SAVEAREA	SAVEREGS	SAVERETN	SAVER5	SAVER6	SAVEWRK1	SAVEWRK2	SAVEWRK3	SAVEWRK4	SAVEWRK5	SAVEWRK6	SAVEWRK7	SAVEWRK9
	SAVFPRES	SAVGREGS	SAVKEYS	SAVPSW	SAVTABLE	SEGINV	SEGPAGE	SHRBPNT	SHRFLAG	SHRFPNT	SHRNAME	SHRNOPRT	SHRPAGE
	SHRSEGCT	SHRSEGNM	SHRTABLE	SHRTSIZE	SHRUSECT	SWPAPP	SWPCHG1	SWPCYL	SWPFLAG	SWPFLAG2	SWPKEY1	SWPPAG	SWPSHR
	SWPTABLE	SWPVM	SYSCYL	SYSFLAG	SYSHRSEG	SYSNAME	SYSPAGCT	SYSPAGLN	SYSPAGNM	SYSPNT	SYSPROT	SYSSEGLN	SYSSIZE
	SYSSTART	SYSTBL	SYSTEM	SYSVADDR	SYSVOL	TRANMODE	TYP2314	TYP3330	TYP3350	VDEVBLOK	VDEVREAL	VDEVRELN	VMABLOK
	VMAFPNT	VMANAME	VMASHRBK	VMASIZE	VMASSIST	VMBLOK	VMCOMND	VMECEXT	VMESTAT	VMEXTCM	VMFPRS	VMGPRS	VMIOWAIT
	VMLOGOFF	VMMLEVEL	VMNSHR	VMOSTAT	VMPA2APL	VMPSTAT	VMPSW	VMQSTAT	VMRSTAT	VMSEG	VMSHR	VMSHRPRC	VMSHRSYS
	VMSIZE	VMSTOR	VMV370R	VSYSRES	XRIGHT16	X40FFS							
DMKCFH	AFREE	AFRET	APTRAN	APTRLK	AQCNWT	ASYSVM	BRING	C1	C14	C15	C2	DEFER	DMKCVTBH
	DMKERMSG	DMKFREE	DMKFRET	DMKPTRAN	DMKPTRUL	DMKQCNRD	DMKQCNWT	DMKRPAGT	DMKRPAPT	DMKSCNFD	DMKSCNVS	DMKSCNVU	DMKSNTBL
	DMKVMAS1	ECBLOK	EDIT	ERRMSG	EXTMASK	F0	F1	F2	F256	F3	F4	F4096	F8
	LOCK	NORET	NOTRESP	PSA	RDEVBLOK	RDEVCODE	RDEVFLAG	RDEVOWN	RDEVTYPE	R0	R1	R10	R11
	R12	R13	R14	R15	R2	R3	R4	R5	R6	R7	R8	R9	SAVCREGS
	SAVEAREA	SAVEREGS	SAVEWRK1	SAVEWRK2	SAVEWRK4	SAVEWRK5	SAVEWRK6	SAVEWRK7	SAVEWRK9	SAVFPRES	SAVGREGS	SAVKEYS	SAVPSW
	SAVTABLE	SEGINV	SHRBPNT	SHRFPNT	SHRNAME	SHRTABLE	SWPFLAG	SWPKEY1	SYSCYL	SYSNAME	SYSPAGCT	SYSPAGLN	SYSPAGNM
	SYSPNT	SYSSIZE	SYSSTART	SYSTBL	SYSTEM	SYSVADDR	SYSVOL	TYP2305	TYP2314	TYP3330	TYP3340	TYP3350	UCASE
	VDEVBLOK	VDEVREAL	VDEVRELN	VMBLOK	VMECEXT	VMLOGOFF	VMPEND	VMPSTAT	VMPSW	VMRSTAT	VMSEG	VMSIZE	VMSTOR
	VMVCR0	VMV370R	VSYSRES										
DMKCFM	ADSPCH	AFREE	AFRET	APSTAT1	AQCNWT	ATTN	BALRSAVE	BLANKS	BUFCNT	BUFFER	BUFINLTH	BUFNXT	BUFSIZE

MODULE	EXTERNAL REFERENCES (LABELS AND MODULES)												
	CLASGRAF	CLASTERM	CONREAD	CPEXADD	CPEXBLOK	CPEXREGS	CPEXSIZE	CPINITD	DMKCFCMD	DMKDSPB	DMKDSPCH	DMKFREE	DMKFRET
	DMKQCNRD	DMKQCNWT	DMKSCHRT	DMKSCNFD	DMKSTKCP	DMKVIOMK	EDIT	ERRMSG	IOMASK	LOCK	NOAUTO	NORET	NOTIME
	PSA	RDEVBLOK	RDEVTYPC	RDEVTYPE	R0	R1	R11	R12	R13	R14	R15	R2	R3
	R4	R5	R6	R7	R8	R9	SAVEAREA	SAVEREGS	SAVER11	SAVER2	SAVEWRK6	TREXLOCK	TREXT
	TREXTERM	TRQBLOK	TRQBSIZE	TRQBTOD	TRQBVAL	TYPBSC	UCASE	VCHADD	VCHELOK	VCHCUINT	VCHCUTBL	VCUADD	VCUBLOK
	VCUDVINT	VCUDVTBL	VDEVADD	VDEVBLOK	VDEVBUSY	VDEVCHBS	VDEVINTS	VDEVPEND	VDEVSTAT	VMBLOK	VMCF	VMCFREAD	VMCFRUN
	VMCFWAIT	VMCHSTRT	VMCHTBL	VMCLASSA	VMCLASSB	VMCLASSC	VMCLASSD	VMCLASSE	VMCLASSF	VMCLASSG	VMCLASSH	VMCPWAIT	VMCUSTRT
	VMDELAY	VMDVSTRT	VMIOINT	VMIOPND	VMKILL	VMLOGOFF	VMLOGON	VMMLEVEL	VMMSTMP	VMOSTAT	VMPEND	VMPRIDSP	VMPSW
	VMQSTAT	VMRSTAT	VMSLEEP	VMSTKO	VMSYSOP	VMTERM	VMTREXT	VMVIRCF	VMVTERM	WAIT			
DMKCFO	ACORETBL	ADSPCH	AFREE	AFRET	AMCHAREA	APSTAT1	APUOPER	AQCNWT	ASYSLC	BLANKS	BUFFER	BUFNXT	CLASTAPE
	CLASURO	CORFLAG	CORRSV	CORTABLE	CPASTAVL	CPASTON	CPCREG6	CPEXADD	CPEXBLOK	CPEXR2	CPEXSIZE	CPMICAVL	CPMICON
	CPSTAT2	C6	DMKCVTBH	DMKCVTDB	DMKCVTDT	DMKCVTHB	DMKDMPAU	DMKDMPDV	DMKDMPSW	DMKDSPCH	DMKDSPNP	DMKERMSG	DMKFREE
	DMKFRET	DMKIOEIR	DMKLOKSW	DMKMCHMS	DMKPTRRC	DMKPTRRL	DMKPTRRU	DMKQCNRD	DMKQCNWT	DMKSCHAP	DMKSCHAU	DMKSCHDL	DMKSCHQ1
	DMKSCHQ2	DMKSCNAU	DMKSCNFD	DMKSCNRA	DMKSCNRU	DMKSTKOP	DMKSYSDT	DMKSYSDW	DMKSYSLG	DMKSYSLW	DMKSYSRV	DMKSYSTM	EDIT
	F1	F2	F3	F4	F5	F7	F8	IPUADDR	IPUADDRX	IRMAND	IRMBIT1	IRMBIT2	IRMBLOK
	IRMBYT1	IRMBYT2	IRMFLG	IRMLNT	IRMOR	IRMRLADD	IRMSIZE	LOCK	MCHAREA	MCHMODEL	MOD3033	NOAUTO	NORET
	PROCIO	PSA	RCHBLOK	RCUBLOK	RCUCHA	RCUCHAOF	RCUCHBOF	RCUCHCOF	RCUCHD	RCUCHDOF	RCUDISA	RCUSTAT	RDEVBLOK
	RDEVDED	RDEVDISA	RDEVFLAG	RDEVIRM	RDEVSTAT	RDEVSYS	RDEVTYPC	RDEVTYPE	RDEVUSER	R0	R1	R10	R11
	R12	R13	R14	R15	R2	R3	R4	R5	R6	R7	R8	R9	SAVEAREA
	SAVEREGS	SAVER11	SAVEWRK1	SAVEWRK2	SAVEWRK3	SAVEWRK4	SAVEWRK5	SAVEWRK7	SAVEWRK8	SVMNOUPD	SYSLOCS	TIMEDISP	TYPPRT
	UCASE	VMAEX	VMAEXP	VMBLOK	VMCLASSA	VMCLASSB	VMCLASSC	VMCLASSD	VMCLASSE	VMCLASSF	VMCLASSG	VMHIPRI	VMIDLE
	VMMCPAST	VMMCR6	VMPNT	VMPSTAT	VMQLEVEL	VMRPAGE	VMRSTAT	VMSTOR	VMUPRIOR	VMUSER	X40FFS	ZEROES	
DMKCFP	ADSPCH	AFREE	AFRET	APSTAT1	APUOPER	ASYSVM	AVMREAL	CC	CHBWAIT	CHXBLOK	CHXCNCT	CHXFLAG	CLASDASD
	CLASGRAF	CLASSPEC	CLASTAPE	CLASTERM	CLASURI	CLASURO	CPEXADD	CPEXBLOK	CPEXFPNT	CPEXMISC	CPEXR0	CPEXR11	CPEXSIZE
	CPRUN	CPSTATUS	CUE	DE	DELPAGES	DMKBLDRL	DMKELDRT	DMKDIADR	DMKDSPCH	DMKFREE	DMKFRET	DMKIOSHA	DMKIOSQR
	DMKIOSRW	DMKLOCKD	DMKLOCKQ	DMKLOKSW	DMKPERT	DMKPGSPO	DMKPGSPP	DMKPTRPW	DMKQCNSY	DMKSCHRT	DMKSCNVD	DMKSSSCF	DMKSTKCP
	DMKSTKIO	DMKTRCPB	DMKUNTFR	DMKVATBC	DMKVCARD	DMKVDREL	DMKVICIN	DMKVIOMK	DMKVMCUA	DMKVSPCO	DMKVSPCR	ECBLOK	EXTCCTRQ
	EXTCPTMR	EXTCPTRQ	EXTCR0	EXTCR14	EXTCR15	EXTCR2	EXTCR4	FFS	F1	IOBCAW	IOBCC3	IOBCSW	IOBFATAL
	IOBFLAG	IOBHIO	IOBHVC	IOBIOER	IOBIRA	IOBLINK	IOBLOK	IOBMISC	IOBMISC2	IOBRES	IOBSIZE	IOBSPEC	IOBSTAT
	IOBTIO	IOBUNSL	IOBUSER	IOBVADD	IOERBLOK	IOEREXT	IOERSIZE	KEEPSEGS	LOCK	MSSPRES	NEWPAGES	NEWSEGS	OLDVMSEG
	PGBLOK	PGBSIZE	PGPNT	PROBSTRT	PROBTIME	PROCIO	PSA	PSAMSS	RCWCCNT	RCWCCW	RCWHEAD	RCWRCNT	RCWTASK
	RDEVAIOB	RDEVBLOK	RDEVCLAS	RDEVFTR	RDEVIOER	RDEVTYPE	RUNUSER	R0	R1	R10	R11	R12	R13
	R14	R15	R2	R3	R4	R5	R6	R7	R8	R9	SAVEAREA	SAVEREGS	SAVER11
	SAVER8	SAVEWRK6	SAVEWRK8	SILI	SUSPEND	SYSVIRT	TIMEDISP	TRQBFPNT	TRQELOK	TRQBQUE	TRQBVAL	TYPCTCA	TYP3210
	TYP3215	TYP3330	TYP3851	VCHADD	VCHBLOK	VCHBUSY	VCHCEDEV	VCHCEPND	VCHCUINT	VCHCUTBL	VCHDED	VCHSTAT	VCONCTL
	VCONRBSZ	VCONRBUF	VCONWBSZ	VCONWBUF	VCUACTV	VCUADD	VCUBLCK	VCUBUSY	VCUCEPND	VCUCHBSY	VCUCUEPN	VCUDVINT	VCUDVTBL
	VCUINTS	VCUSTAT	VDEVADD	VDEVAUCR	VDEVBLOK	VDEVBUSY	VDEVCCW1	VDEVCFLG	VDEVCHAN	VDEVCHBS	VDEVCON	VDEVCPEX	VDEVCSPL
	VDEVCSW	VDEVCUE	VDEVDED	VDEVDIAL	VDEVENAB	VDEVFEED	VDEVFLAG	VDEVFLG2	VDEVINTS	VDEVIOB	VDEVIOER	VDEVLINK	VDEVNRDY
	VDEVODE	VDEVPEND	VDEVREAL	VDEVRES	VDEVRRB	VDEVRRF	VDEVRSRL	VDEVSFLG	VDEVSPL	VDEVSTAT	VDEVTYPC	VDEVTYPE	VDEVUC
	VDEVUSER	VIRTUAL	VMBCAUTH	VMBLOK	VMCHSTRT	VMCHTBL	VMCPUTMR	VMCUSTRT	VMCXSTAT	VMDSTAT	VMDVSTRT	VMECEXT	VMESTAT
	VMEXWAIT	VMFAUTO	VMFSTAT	VMIDLE	VMINQ	VMINVPAG	VMIOACTV	VMIOINT	VMIOLOG	VMIOPND	VMIOWAIT	VMKILL	VMLOGOFF
	VMMICSVC	VMNOTRAN	VMOSTAT	VMPAGEX	VMPEND	VMPGPND	VMPGPNT	VMPGWAIT	VMPSTAT	VMPSW	VMPXINT	VMRSTAT	VMSEG
	VMSIZE	VMSTOR	VMTIO	VMTLEVEL	VMTMOUTQ	VMTRBRIN	VMTRCTL	VMUSER	VMVCR0	VMVCR14	VMVTERM	VMV370R	VRRBLOK
	VRRCPEX	VRRRES	VRRSTAT	WAIT	XINTBLOK	XINTNEXT	XINTSIZE	XINTSORT	XRIGHT24	ZEROES			

MODULE	EXTERNAL REFERENCES (LABELS AND MODULES)												
DMKCFS	AFREE	AFRET	APSTAT1	APUOPER	AQCNWT	AVMREAL	BLANKS	BUFCNT	BUFFER	BUFNXT	CPASTAVL	CPMICAVL	CPMICON
	CPSTAT2	C1	DMKBLDEC	DMKCFPRR	DMKCVTBH	DMKCVTDB	DMKCVTHB	DMKERMSG	DMKFREE	DMKFRET	DMKQCNWT	DMKSCHRT	DMKSCH80
	DMKSCNAU	DMKSCNFD	DMKUDRFU	DMKUDRMD	DMKUDRRV	ECBLOK	EXTCCTRQ	EXTCPTRQ	EXTSIZE	FFS	F2	F3	F7
	F8	IPUADDR	LOCK	LPUADDR	LPUADDRX	MICBLOK	MICCREG	MICEVMA	MICRSEG	MICSIZE	MICVPSW	MICVTMR	MICWORK
	NORET	PREFIXB	PSA	R0	R1	R10	R11	R12	R13	R14	R15	R2	R3
	R4	R5	R6	R7	R8	R9	SAVEAREA	SAVEREGS	SAVEWRK1	SAVEWRK2	SAVEWRK4	SAVEWRK5	SAVEWRK6
	SAVEWRK7	SAVEWRK8	TEMPSAVE	TIMER	TRQBIRA	TRQBLOK	TRQBSIZE	TRQBUSER	UDBFELOK	UDBFSIZE	UDBFVADD	UDIRBLOK	UDIRDISP
	UMACAFF	UMACBLOK	VMADSTOP	VMAFF	VMAFFON	VMBLOK	VMCFRUN	VMCLASSA	VMCLASSB	VMCLASSC	VMCLASSD	VMCLASSE	VMCLASSF
	VMCLASSG	VMCLEVEL	VMCPUID	VMECEXT	VMESTAT	VMFAUTO	VMFSTAT	VMFVTMR	VMISAM	VMMACCON	VMMADDR	VMMCODE	VMMCR6
	VMMFE	VMMICRO	VMMICSVC	VMMIMSG	VMMLEVEL	VMMLINED	VMMLVL2	VMMSGON	VMMSVC	VMMTEXT	VMMVTMR	VMM360	VMNOTRAN
	VMOSTAT	VMPAGEX	VMPFUNC	VMPSTAT	VMPSW	VMRON	VMSEG	VMSPMFLG	VMSTMPI	VMTIMER	VMTLEVEL	VMTON	VMTRCTL
	VMTREX	VMTRQBLK	VMUSER	VMVCR0	VMV370R	VMWNGON	ZEROES						
DMKCFT	APSTAT1	ASYSLC	CLASGRAF	CLASSPEC	CLASTERM	DMKCVTBH	DMKCVTDB	DMKERMSG	DMKSCNFD	DMKSCNVD	DMKSYSCD	DMKSYSES	DMKSYSLD
	DMKSYSLE	DMKTBLGR	F1	F2	F255	F4095	LOCK	NICAPL	NICATCF	NICBLOK	NICFLAG	NICLLEN	NICPSUP
	NICSIZE	NICTEXT	NICTMCD	PROCIO	PSA	RDEVAPLP	RDEVATOF	RDEVBLOK	RDEVFLAG	RDEVLLEN	RDEVNICL	RDEVPSUP	RDEVTEXT
	RDEVTFLG	RDEVTMCD	RDEVTYPC	RDEVTYPE	R0	R1	R10	R11	R12	R13	R14	R15	R2
	R3	R4	R5	R6	R7	R8	SAVEAREA	SAVEREGS	SAVEWRK1	SYSLOCS	TYPBSC	TYPTTY	TYP3277
	TYP3278	VMBLOK	VMDVSTRT	VMGRFTAB	VMMCPENV	VMMLEVEL	VMMSTMP	VMTCDEL	VMTERM	VMTESCP	VMTLDEL	VMTLEND	VMTRMID
	VMVTERM	X40FFS	ZEROES										
DMKCKP	ACCTBLOK	ACCTUSER	ACNTBLOK	ACNTCCW	ACNTCODE	ACNTCONT	ACNTDATA	ACNTDEVC	ACNTIOCT	ACNTNCYL	ACNTNEXT	ACNTNUM	ACNTPGRD
	ACNTSTOP	ACNTTIME	ACNTUSER	ACNTVTIM	ALARM	ARIOCC	ARIOCH	ARIOCT	ARIOCU	ARIODV	ARIOPR	ARIOPU	ARIORD
	ARSPPR	ASYSLC	ASYSVM	ATTN	BUSY	CAW	CC	CE	CLASDASD	CLASGRAF	CLASSPEC	CLASTAPE	CLASTERM
	CLASURI	CLASURO	CPCREG0	CPID	CSW	CUE	C0	C2	C3	DATE	DE	DEVCARD	DMKOPRWT
	DMKRSPAC	DMKRSPCV	DMKRSPDL	DMKRSPHQ	DMKRSPID	DMKRSPMN	DMKRSPPR	DMKRSPPU	DMKRSPRD	DMKSAV	DMKSAVRS	DMKSYSCK	DMKSYSDT
	DMKSYSLG	DMKSYSOC	DMKSYSOW	DMKSYSRM	DMKSYSTP	DMKSYSWM	DMKTMRPT	ERRCCW	FLAG	FTR35MB	F1	INTPR	INTTIO
	IONPSW	IOOPSW	IPLPSW	LOCK	MCNPSW	NICBLOK	NICDISA	NICDISB	NICENAB	NICFLAG	NICLGRP	NICLINE	NICSIZE
	NICSTAT	NICTERM	NICTYPE	OWNDLIST	OWNDRDEV	PRNPSW	PROPSW	PSA	RCHADD	RCHBLOK	RCHCUTBL	RCUADD	RCUBLOK
	RCUCHA	RCUDVTBL	RCUPRIME	RCUSUB	RCUTYPE	RDEVACNT	RDEVADD	RDEVAIOB	RDEVAUTO	RDEVBLOK	RDEVCLAS	RDEVCUA	RDEVDED
	RDEVDELP	RDEVDISA	RDEVDISB	RDEVDRAN	RDEVENAB	RDEVFLAG	RDEVFSEP	RDEVFTR	RDEVIMAG	RDEVLCEP	RDEVLNCP	RDEVMAX	RDEVMDL
	RDEVNCP	RDEVNICL	RDEVRECS	RDEVSEP	RDEVSPL	RDEVSTAT	RDEVTMAT	RDEVTYPC	RDEVTYPE	RDEVXSEP	RECBLOK	RECCYL	RECMAX
	RECPNT	RECSIZE	RECUSED	RSPLCTL	RSPSFBLK	R0	R1	R10	R11	R12	R13	R14	R15
	R2	R3	R4	R5	R6	R7	R8	R9	SAVEWRK2	SEEK	SETUP	SETUP1	SETUP2
	SFBCLAS	SFBCOPY	SFBDATE	SFBDIST	SFBFILID	SFBFIRST	SFBFLAG	SFBFLAG2	SFBFNAME	SFBFTYPE	SFBLAST	SFBLOK	SFBORIG
	SFBPNT	SFBPURGE	SFBRECER	SFBRECS	SFBSIZE	SFBSTART	SFBUSER	SHQBSIZE	SILI	SIZE	SKIP	STARTIME	STCODE
	SUSPEND	SYSLOCS	TODATE	TYPBSC	TYPE	TYPPRT	TYPPUN	TYP2305	TYP2314	TYP3210	TYP3277	TYP3278	TYP3284
	TYP3330	TYP3340	TYP3350	TYP3705	TYP3800	TYP3851	UC	USERCARD	VCHBLOK	VCHCUTBL	VCUBLOK	VCUDVTBL	VDEVBLOK
	VDEVBND	VDEVCLAS	VDEVCOPY	VDEVDED	VDEVEXTN	VDEVFLAG	VDEVREAL	VDEVSFLG	VDEVSPL	VDEVSTAT	VDEVTDSK	VDEVTMAT	VDEVTYPC
	VDEVTYPE	VDEVXFER	VMACNT	VMACOUNT	VMBLOK	VMCHSTRT	VMCHTBL	VMCUSTRT	VMDIST	VMDVSTRT	VMIOCNT	VMLOGON	VMPGREAD
	VMPNT	VMRSTAT	VMTIMEON	VMTTIME	VMUSER	VSPLCTL	VSPSFBLK	VSPXBLOK	VSPXXUSR				
DMKCKS	ACTSFB	ADDSFB	AFREE	AFRET	APSTAT1	APTRAN	APTRLK	ARIODV	ARSPPR	ARSPPU	ARSPRD	ASYSVM	BRING
	CHGSFB	C1	DEFER	DELSFB	DMKCVTBD	DMKERMSG	DMKFREE	DMKFRET	DMKLOCKD	DMKLOCKQ	DMKLOCKT	DMKPGTTU	DMKPGTVG
	DMKPGTVR	DMKPTRAN	DMKPTRUL	DMKQCNSY	DMKQNTBL	DMKRPAGT	DMKRPAPT	DMKRSPHQ	DMKRSPID	DMKSCNRD	DMKSCNRU	DMKSYSCH	DMKSYSCN

MODULE	EXTERNAL REFERENCES (LABELS AND MODULES)												
	DMKSYSOC	DMKSYSOW	FFS	F1	F10	F24	F255	F3	F4	LOCK	MSGTYPE	NPRCNT	NPRNAME
	NPRPNT	NPRTBL	OPNSFB	OWNDLIST	OWNDRDEV	OWNDVSER	PCHCHN	PROCIO	PRTCHN	PSA	RDEVALLN	RDEVBLOK	RDEVCLAS
	RDEVCODE	RDEVDISA	RDEVDRAN	RDEVFLAG	RDEVFSEP	RDEVIMAG	RDEVPREF	RDEVRECS	RDEVSER	RDEVSPL	RDEVSTAT	RDEVTYPE	RDEVXSEP
	RDRCHN	RECBLOK	RECCYL	RECMAP	RECMAX	RECPNT	RECSIZE	RECUSED	R0	R1	R10	R11	R12
	R13	R14	R15	R2	R3	R4	R5	R6	R7	R8	R9	SAVEAREA	SAVEREGS
	SAVER1	SAVER2	SAVER8	SAVEWRK1	SAVEWRK2	SAVEWRK3	SAVEWRK4	SAVEWRK5	SAVEWRK6	SAVEWRK7	SAVEWRK8	SAVEWRK9	SFBCOPY
	SFBDIST	SFBDUMP	SFBEOF	SFBFILID	SFBFLAG	SFBFLAG2	SFBINUSE	SFBLAST	SFBLOK	SFBMON	SFBOPEN	SFBPNT	SFBRECER
	SFBRECNO	SFBRECS	SFBRSTRT	SFBSIZE	SFBSTART	SFBTIME	SFBUHCLD	SHQBLOK	SHQBSIZE	SHQUSER	SPFILID	SPLINK	SPNXTPAG
	SPPREPAG	SPRECNUM	SPTIME	SYSIPLDV	SYSTEM	TYP2314	TYP3330	TYP3350	TYP3800	VMBLOK	VMSEG	ZEROES	
DMKCLK	AEXTSP	ALARM	APSTAT1	APSTAT4	APUOPER	AQCNWT	ASYSOF	BLKMPX	CKCMASK	CPCREG0	CPSTATUS	CPTMASK	CPWAIT
	C0	DMKPSADU	DMKQCNWT	EMSMASK	EMSPCLKC	EMSPEND	EMSPSYNC	EMSRCLKC	EMSREC	EMSRSYNC	EXTMODE	INTMASK	IPUADDRX
	KEYMASK	LOCK	MCHEK	MFAMASK	NOTIME	OPERATOR	PAGE4K	PROCIO	PROCSCHK	PSA	R0	R1	R11
	R12	R13	R14	R15	R2	SAVEAREA	SAVEREGS	SIGCLK	SIGEMS	SIGQUI	SIGRES	SIGSYNC	SIGXC
	SYNCMASK	TODSYNC	XCMASK	ZEROES									
DMKCNS	ADSPCH	AFREE	AFRET	ALARM	APSTAT1	APTRAN	APUOPER	ASYSVM	ATTN	BALRSAVE	BALR3	BALR6	BALR9
	BLANKS	BRING	BUSY	CAW	CC	CCC	CD	CDC	CE	CHC	CLASTERM	CMDREJ	CODE
	CONACTV	CONADDR	CONCCW1	CONCCW2	CONCCW3	CONCCW4	CONCNT	CONCNTL	CONCOMND	CONDATA	CONESCP	CONFLAG	CONOUTPT
	CONPARM	CONPNT	CONRESP	CONRETN	CONRTRY	CONSPLT	CONSTAT	CONSYNC	CONTASK	CONTSIZE	CONTSKSZ	CONUSER	CPEXADD
	CPEXBLOK	CPEXREGS	CPEXR10	CPEXSIZE	CPID	CSW	C1	DATACHK	DE	DEFER	DMKBLDVM	DMKCFMAT	DMKCFMBK
	DMKCPIEM	DMKCVTBH	DMKDSPCH	DMKERMSG	DMKFREE	DMKFRET	DMKIOERR	DMKIOEST	DMKIOSQR	DMKLOKSW	DMKMSWR	DMKPTRAN	DMKQCNCL
	DMKQCNET	DMKQCNTO	DMKSCNRD	DMKSCNRU	DMKSTKMP	DMKTBLCI	DMKTBLCO	DMKTBLPI	DMKTBLPO	DMKTBLTI	DMKTBLTO	DMKTBLUP	DMKTBMMI
	DMKTBMMO	DMKTBMNI	DMKTBMNO	DMKTRMID	EDIT	FFS	F1	F10	F15	F16	F2	F256	F4
	F8	IFCC	IL	INHIBIT	INTREQ	IOBCAW	IOBCC1	IOBCC3	IOBCSW	IOBERP	IOBFATAL	IOBFLAG	IOBIOER
	IOBIRA	IOBLINK	IOBLOK	IOBMISC	IOBRADD	IOBRES	IOBSIZE	IOBSPEC	IOBSTAT	IOBUNSL	IOBUSER	IOERBLOK	IOERDATA
	IOEREXT	IOERFLG3	IOERNUM	IOERREAD	IOERSIZE	LOCK	LOGDRCP	LOGHOLD	MSGNUM	NOAUTO	PREFIXA	PRGC	PRIORITY
	PROCIO	PRTC	PSA	RDEVACTV	RDEVAIOB	RDEVATNC	RDEVATOF	RDEVBLOK	RDEVCON	RDEVCORR	RDEVCTL	RDEVDISA	RDEVDISB
	RDEVDROP	RDEVENAB	RDEVEPMD	RDEVFLAG	RDEVHIO	RDEVIDNT	RDEVICER	RDEVLOG	RDEVNRDY	RDEVPREP	RDEVPSUP	RDEVPTTC	RDEVRCNT
	RDEVREST	RDEVSADN	RDEVSTAT	RDEVSTA2	RDEVSYNC	RDEVTFLG	RDEVTMCD	RDEVTYPC	RDEVTYPE	RDEVUSC8	RDEVUSER	RETRYSW	R0
	R1	R10	R11	R12	R13	R14	R15	R2	R3	R4	R5	R6	R7
	R8	R9	SAVEAREA	SAVER0	SAVER1	SAVER2	SILI	SKIP	SM	SVMSTAY	SVMUNLOK	SYSTEM	TEMPR0
	TEMPSAVE	TIMEDISP	TRACBEF	TRACCURR	TRACEND	TRACFLG2	TRACPROC	TRACSTRT	TYPTTY	TYPUNDEF	TYP1050	TYP2741	TYP3210
	UC	UCASE	UE	VMBLOK	VMCF	VMCFWAIT	VMLOGOFF	VMLOGON	VMMCPENV	VMMLEVEL	VMOSTAT	VMRSTAT	VMSEG
	VMSYSOP	VMTCDEL	VMTERM	VMTLEND									
DMKCPB	ADSPCH	AFREE	AFRET	APTRAN	AQCNWT	ASYSVM	BLANKS	BRING	CLASSPEC	CLASTAPE	CLASTERM	CLASURI	CLASURO
	CPEXBLOK	CPEXMISC	C1	DE	DEFER	DEVICE	DMKCFPRD	DMKCFPRR	DMKCVTBH	DMKCVTHB	DMKDSPCH	DMKERMSG	DMKFREE
	DMKFRET	DMKIOSQR	DMKIOSRW	DMKPGSPO	DMKPTRAN	DMKQCNWT	DMKSCNFD	DMKSCNVU	DMKSSSMQ	DMKSTKCP	DMKVATBC	DMKVATMD	DMKVIOMK
	ERROR	EXTMODE	FFS	F16	F3	F4	F6	IOBCAW	IOBIRA	IOBLOK	IOBMISC	IOBSIZE	IOBUSER
	LOCK	MSSNEXT	MSSTASK2	MSSUSER	NORET	PSA	RCHBLCK	RCUBLOK	RCUCHA	RCUPRIME	RCUSUB	RCUTYPE	RCWCCNT
	RCWCCW	RCWHEAD	RCWTASK	RDEVBLOK	RDEVBUSY	RDEVCUA	RDEVSTAT	R0	R1	R10	R11	R12	R13
	R14	R15	R2	R3	R4	R5	R6	R7	R8	SAVEAREA	SAVEREGS	SAVERETN	SAVEWRK2
	SAVEWRK4	SAVEWRK5	SILI	SYSTEM	TRANMODE	TYPCTCA	TYP3210	VCHADD	VCHBLOK	VCHCUINT	VCUADD	VCUBLOK	VCUDVINT
	VDEVADD	VDEVBLOK	VDEVBUSY	VDEVDED	VDEVINTS	VDEVNRDY	VDEVPEND	VDEVREAL	VDEVSTAT	VDEVTYPC	VDEVTYPE	VMBLOK	VMESTAT
	VMEXTCM	VMIOINT	VMIOPND	VMPA2APL	VMPEND	VMPSTAT	VMPSW	VMPXINT	VMQSTAT	VMSEG	VMUSER	VMV370R	XINTBLOK

MODULE	EXTERNAL REFERENCES (LABELS AND MODULES)												
	XINTCODE	XINTNEXT	XINTSIZE	XINTSORT	X40FFS	ZEROES							
DMKCPE	LOCK												
DMKCPI	ACORETBL	ADSPCH	AEXTSP	AFREE	AFRET	ALARM	ALOCBLOK	ALOCCYL1	ALOCCYL2	ALOCMAP	ALOCMAX	ALOCNTMP	ALOCPNT
	ALOCUSED	APAGCP	APSTAT1	APTRAN	APTRLK	APUNONLN	APUOPER	AQCNWT	ARIOCH	ARIOCT	ARIOCU	ARIODV	ASYSOP
	ASYSVM	AUTGO	BALRSAVE	BALR0	BALR1	BALR14	BALR2	BALR6	BALR8	BLKMPX	BRING	BUFCNT	BUFFER
	BUFIN	BUFNXT	BUFSIZE	BUSY	CAW	CC	CCC	CE	CHNGMSG	CKCMASK	CLASDASD	CLASGRAF	CLASSPEC
	CLASTAPE	CLASTERM	CLASURO	CODE	CORCFLCK	CORCP	CORFLAG	CORFPNT	CORFREE	CORIOLCK	CORLCNT	CORPGPNT	CORSWPNT
	CORTABLE	CPASTAVL	CPASTON	CPCREG0	CPCREG6	CPEXSIZE	CPID	CPINITD	CPMICAVL	CPMICON	CPSTATUS	CPSTAT2	CPSUPER
	CPUID	CPULOG	CPUMODEL	CPWAIT	CSW	CUE	C0	C1	C14	C2	C6	DAMAGRPT	DATE
	DE	DEFER	DMKALGON	DMKAPIPR	DMKBLDRT	DMKCCWB1	DMKCCWB2	DMKCCWB3	DMKCCWB4	DMKCCWB5	DMKCCWB6	DMKCCWB7	DMKCCWB8
	DMKCCWGN	DMKCCWL1	DMKCCWL2	DMKCCWL3	DMKCCWL4	DMKCCWL5	DMKCCW0	DMKCCW1	DMKCFMEN	DMKCKP	DMKCLKCK	DMKCNSEN	DMKCPEID
	DMKCPEML	DMKCPEND	DMKCPVAE	DMKCQRFI	DMKCSOSD	DMKCVTBD	DMKCVTBH	DMKCVTDT	DMKDMPAA	DMKDMPAU	DMKDMPDT	DMKDMPDV	DMKDMPMA
	DMKDMPRC	DMKDMPSA	DMKDMPSD	DMKDMPSF	DMKDMPTD	DMKDSPCH	DMKDSPNP	DMKDSP0	DMKDSP1	DMKDSP2	DMKFREAP	DMKFREE	DMKFREHI
	DMKFRELG	DMKFRELO	DMKFRESV	DMKFRET	DMKFRETR	DMKHVDPP	DMKIOEFL	DMKIOSIN	DMKIOSMQ	DMKIOSQR	DMKLOGOP	DMKMCHIN	DMKMNIST
	DMKNETAE	DMKNLDR	DMKPAGHI	DMKPAGLO	DMKPAGST	DMKPGTBN	DMKPGTPG	DMKPGTP0	DMKPGTP4	DMKPGTP5	DMKPGTTM	DMKPGTTU	DMKPGTT0
	DMKPGTT4	DMKPGTT5	DMKPGT4P	DMKPGT4T	DMKPGT5P	DMKPGT5T	DMKPGT90	DMKPRGIN	DMKPSADU	DMKPSAEX	DMKPTRAN	DMKPTRCP	DMKPTRFA
	DMKPTRFN	DMKPTRF1	DMKPTRLK	DMKPTRRM	DMKPTRUL	DMKQCNRD	DMKQCNWT	DMKRIOCN	DMKRIORN	DMKRPAPT	DMKSAV	DMKSCHLI	DMKSCHMD
	DMKSCHQ1	DMKSCHQ2	DMKSCHST	DMKSCHTI	DMKSCNRA	DMKSCNRD	DMKSCNRU	DMKSCNVS	DMKSCNVU	DMKSVCHI	DMKSVCIN	DMKSVCLO	DMKSVCNS
	DMKSYM	DMKSYMTB	DMKSYSAP	DMKSYSAT	DMKSYSDU	DMKSYSDW	DMKSYSFP	DMKSYSNU	DMKSYSOC	DMKSYSOW	DMKSYSRM	DMKSYSRV	DMKSYSTE
	DMKSYSTI	DMKSYSTR	DMKSYSTS	DMKSYSTZ	DMKSYSUD	DMKSYSUR	DMKSYSVL	DMKUDRBV	DMKUDRFU	DMKUNTFR	DMKUNTRN	DMKVATZP	DMKVATZS
	DMKVMASH	DMKVMI	DMKWRMST	EDIT	EXNPSW	EXTMASK	EXTMODE	FFS	FTRRPS	FTRRSRL	FTRVIRT	FTR35MB	FTR70MB
	F0	F1	F10	F15	F2	F240	F3	F4	F4095	F4096	F5	F7	F8
	F9	HARDSTOP	IDLEWAIT	IFCC	INTMASK	INTREQ	IOBCAW	IOBIRA	IOBLCK	IOBMSIZE	IOBSIZE	IOBUSER	IONPSW
	IONTWAIT	IPLCCW1	IPLPSW	IPUADDR	IPUADDRX	KEYMASK	LOCK	LPUADDR	LPUADDRX	MCHEK	MCNPSW	MFAMASK	MPFEAT
	MSSPRES	NEWPAGES	NEWSEGS	NICBLOK	NICDISA	NICNAME	NICSIZE	NICSTAT	NOAUTO	NORET	NOTIME	OPERATOR	OWNDLIST
	OWNDPREF	OWNDRDEV	OWNDVSER	PAGCORE	PAGEWAIT	PAGE4K	PGREAD	PREFIXA	PREFIXB	PRNPSW	PROBTIME	PROCIO	PROPSW
	PSA	PSACPXBP	PSAMSS	PSBCLR2	PSECLR2	PSENDCLR	RCHADD	RCHBLOK	RCHCUTBL	RCUADD	RCUBLOK	RCUCHA	RCUCHAOF
	RCUCHBOF	RCUCHCOF	RCUCHD	RCUCHDOF	RCUDISA	RCUDVTBL	RCUPRIME	RCUSTAT	RCUSUB	RCUTYPE	RDEVADD	RDEVAIOB	RDEVALLN
	RDEVALT	RDEVATOF	RDEVAUTO	RDEVBLOK	RDEVCODE	RDEVCUA	RDEVCUB	RDEVDISA	RDEVENAB	RDEVEXTN	RDEVFLAG	RDEVFTR	RDEVIDNT
	RDEVMAX	RDEVMDL	RDEVNICL	RDEVNRDY	RDEVOWN	RDEVPNT	RDEVPREF	RDEVPTTC	RDEVRUN	RDEVSER	RDEVSTAT	RDEVSTA2	RDEVSYS
	RDEVTFLG	RDEVTMCD	RDEVTYPC	RDEVTYPE	RDEVUSER	RECBLOK	RECCYL	RECMAP	RECMAX	RECPNT	RECSIZE	RECUSED	RSPXSIZE
	RUNCR0	RUNCR1	RUNUSER	R0	R1	R10	R11	R12	R13	R14	R15	R2	R3
	R4	R5	R6	R7	R8	R9	SAVESIZE	SEGPAGE	SETUP	SFBLOK	SFBORIG	SFBSTART	SFBUSER
	SIGIPR	SIGSENSE	SIGWAKE	SIGXC	SILI	SM	STARTIME	SVCNPSW	SWPCHG1	SWPCYL	SWPFLAG	SYNCLOG	SYSIPLDV
	SYSTEM	TEMPR0	TEMPR14	TEMPR15	TEMPR2	TEMPR3	TEMPR4	TEMPR5	TEMPSAVE	TIMEDISP	TIMER	TODATE	TRACCURR
	TRACEFLG	TRACEND	TRACPROC	TRACSTRT	TRQBIRA	TRQBLOK	TRQBSIZE	TRQBTOD	TRQBUSER	TRQBVAL	TYPBSC	TYP2305	TYP2314
	TYP2741	TYP3066	TYP3210	TYP3277	TYP3278	TYP3330	TYP3340	TYP3350	TYP3800	TYP3851	UC	UCASE	UDIRBLOK
	UDIRPASS	VMAPTIME	VMBLOK	VMCPTIME	VMDFTPNT	VMLOGON	VMMFE	VMMSVC	VMPAGES	VMRSTAT	VMSEG	VMSIZE	VMSTOR
	VMTERM	VMTTIME	VMUSER	WAIT	XPAGNUM	XRIGHT16	ZEROES						
DMKCPS	ADSPCH	AEXTSP	AFREE	AFRET	APSTAT1	APTRAN	APTRLK	APUOPER	AQCNWT	ARIOCH	ARIOCT	ARIOCU	ARIODV
	ASYSOP	ASYSVM	BLANKS	BRING	BUFFER	BUFNXT	CPSTOP	CLASDASD	CLASGRAF	CLASTAPE	CLASTERM	CLASURI	CLASURO
	CPCREG8	CPEXADD	CPEXBLOK	CPEXREGS	CPEXR0	CPEXR12	CPEXSIZE	CPID	C1	C8	DE	DEFER	DEVICE
	DFRET	DMKCFCSC	DMKCFMBK	DMKCFPRD	DMKCPUVY	DMKCVTBD	DMKCVTBH	DMKCVTHB	DMKDMPRS	DMKDSBRD	DMKDSBSD	DMKDSPCH	DMKERMSG

MODULE	EXTERNAL REFERENCES (LABELS AND MODULES)												
	DMKFREE	DMKFRET	DMKIOESR	DMKIOSHA	DMKIOSQR	DMKLOKSW	DMKMNISH	DMKPRGMC	DMKPTRAN	DMKPTRLK	DMKPTRUL	DMKQCNWT	DMKRSESD
	DMKSCNFD	DMKSCNNP	DMKSCNRA	DMKSCNRN	DMKSCNRU	DMKSCNVD	DMKSCNVN	DMKSCNVU	DMKSPLDL	DMKSTKCP	DMKSTKIO	DMKTAPRL	DMKVSPCO
	FFS	F2	F3	F7	F8	IOBCC3	IOBCP	IOBCSW	IOBFLAG	IOBHIO	IOBHVC	IOBIMSTK	IOBIOER
	IOBIRA	IOBLINK	IOBLOK	IOBMISC	IOBMISC2	IOBPATHF	IOBRADD	IOBSIZE	IOBSPEC	IOBSTAT	IOBTIO	IOBUNSL	IOBUSER
	IOBVADD	IOERBLOK	IOERDATA	IOEREXT	IOERSIZE	IPUADDRX	LOCK	MONAIOB	MONARDB	MONCOM	MONFLAG1	MONFLAG3	MONIOBF
	MONUSER	NICSIZE	NOADD	NORET	ON	PRIORITY	PROCIC	PSA	RANGE	RCHBLOK	RCHCUTBL	RCUBLOK	RCUCHA
	RCUCHAOF	RCUCHBOF	RCUCHCOF	RCUCHD	RCUCHDOF	RCUDISA	RCUDVTBL	RCUPRIME	RCUSTAT	RCUSUB	RCUTYPE	RDEVADD	RDEVAIOB
	RDEVBLOK	RDEVBUSY	RDEVCTRS	RDEVCUA	RDEVDED	RDEVDELP	RDEVDISA	RDEVDRAN	RDEVENAB	RDEVEPLN	RDEVFIOB	RDEVFLAG	RDEVIOER
	RDEVLCEP	RDEVLNCP	RDEVLNKS	RDEVMAX	RDEVMOUT	RDEVNICL	RDEVNRDY	RDEVOWN	RDEVRCVY	RDEVRSVD	RDEVSCED	RDEVSER	RDEVSPL
	RDEVSTAT	RDEVSYS	RDEVTYPC	RDEVTYPE	RDEVUSER	R0	R1	R10	R11	R12	R13	R14	R15
	R2	R3	R4	R5	R6	R7	R8	R9	SAVEAREA	SAVEREGS	SAVER11	SAVER2	SAVEWRK1
	SAVEWRK2	SAVEWRK3	SAVEWRK4	SAVEWRK6	SAVEWRK7	SAVEWRK8	SAVEWRK9	SIGEMS	SIGQUI	SIGSHD	SPOOLED	SYSTEM	TIMEDISP
	TYPBSC	TYP2305	TYP3705	TYP3800	VDEVBLOK	VDEVCSPL	VDEVFLAG	VDEVSPL	VMBLOK	VMDVSTRT	VMSEG	VMUSER	VMVTERM
	X40FFS	ZEROES											
DMKCPU	ACORETBL	ACTIVTRQ	ADSPCH	AEXTSP	AFREE	AFRET	ALOKVM	AMCHAREA	APSTAT1	APSTAT2	APSTAT4	APTRAN	APTRLK
	APUOPER	AQCNWT	ARIODV	ASYSVM	ATMRSN	BRING	CORCP	CORFLAG	CORFPNT	CORIOLCK	CORPGPNT	CORRSV	CORSHARE
	CORSWPNT	CORTABLE	CORVM	CPCREG0	CPEXADD	CPEXBLOK	CPEXDEFR	CPEXPROC	CPEXREGS	CPEXR1	CPEXR11	CPEXR12	CPEXSIZE
	CPEXTYPE	CPID	CPPTLBR	CPUMCELL	C0	C1	DEFER	DMKAPIPR	DMKCLKCK	DMKCQRWS	DMKCVTBH	DMKDMPAA	DMKDMPMA
	DMKDMPSA	DMKDSPCH	DMKDSPNP	DMKFREAP	DMKFREE	DMKFRET	DMKLOKSW	DMKLOKSY	DMKMCTAF	DMKPAGWS	DMKPGTSP	DMKPTRAN	DMKPTRFR
	DMKPTRFT	DMKPTRPW	DMKPTRRC	DMKPTRSC	DMKPTRUL	DMKQCNWT	DMKSCHCA	DMKSCHTQ	DMKSCNVS	DMKSNTBL	DMKSTKMP	DMKSTKOP	DMKSYSOW
	DMKVMASW	DMKVMAS1	EXTMASK	F1	F15	F16	F255	F256	F4	F4096	F7	F8	IPUADDR
	IPUADDRX	LOCK	LPUADDR	LPUADDRX	MCHAREA	MCHCPEX	MCHFIX	MCHLEN1	MCHREC	MFAMASK	MPFEAT	NORET	OWNDLIST
	OWNDRDEV	PAGBMP	PAGCORE	PAGECUR	PAGERATE	PAGINVAL	PAGTABLE	PAGTOT	PAGTSWP	PGWAITIM	POFFLINE	PREFIXA	PREFIXB
	PROCIO	PSA	PSACPXBP	PWTPAGES	RDEVBLOK	RDEVCODE	RDEVTYPE	R0	R1	R10	R11	R12	R13
	R14	R15	R2	R3	R4	R5	R6	R7	R8	R9	SAVEAREA	SAVEREGS	SAVER1
	SAVER11	SAVEWRK1	SAVEWRK2	SAVEWRK3	SAVEWRK4	SAVEWRK5	SAVEWRK6	SAVEWRK7	SAVEWRK8	SAVEWRK9	SHRFLAG	SHRFPNT	SHRNAME
	SHRNOPRT	SHRPAGE	SHRSEGCT	SHRSEGNM	SHRTABLE	SIGEMS	SIGIPR	SIGQUI	SIGSENSE	SIGSTOP	SIGWAKE	SIGXC	STACKVM
	START	STOP	SWPCHG1	SWPCODE	SWPCYL	SWPFLAG	SWPRECMP	SWPTABLE	SWPVM	SYSHRSEG	SYSNAME	SYSPAGLN	SYSPAGNM
	SYSPNT	SYSSTART	SYSTBL	SYSTEM	SYSVOL	TIMEDISP	TRACSTRT	TRQBFPNT	TRQBLOK	TRQBVAL	TTSEGCNT	TYPE	TYP2305
	TYP2314	TYP3330	TYP3350	UNSHRVM	VMAFF	VMAFFON	VMBLOK	VMDEFSTK	VMDFTPNT	VMESTAT	VMINVPAG	VMLOCK	VMOSTAT
	VMPAGES	VMPDISK	VMPDRUM	VMPEND	VMPGWAIT	VMPNT	VMRSTAT	VMSEG	VMSHR	VMSHRPRC	VMTTIME	ZEROES	
DMKCPV	ACORETBL	AFREE	APAGCP	APSTAT1	APTRAN	APUOPER	AQCNWT	ARIOCH	ARIOCT	ARIOCU	ARIODV	ASYSVM	AVMREAL
	BALRSAVE	BRING	CLASDASD	CLASGRAF	CLASTERM	CORCFLCK	CORFLAG	CORFPNT	CORRSV	CORSHARE	CORTABLE	CORVM	CPEXADD
	CPEXBLOK	CPEXREGS	CPEXR12	CPEXSIZE	C1	DEFER	DMKACCDV	DMKACOFF	DMKACOTM	DMKCNSEN	DMKCVTBH	DMKCVTHB	DMKDSPNP
	DMKERMSG	DMKFREE	DMKGRFEN	DMKLOKSW	DMKPGSPR	DMKPTRAN	DMKPTRUL	DMKQCNWT	DMKRNH	DMKSCHPU	DMKSCNAU	DMKSCNFD	DMKSCNRD
	DMKSCNRN	DMKSCNRU	DMKSTKCP	F1	F2	F3	F4096	F8	F9	LOCK	NORET	PREFIXB	PROCIO
	PSA	RCHBLOK	RCHCUTBL	RCUBLOK	RCUDVTBL	RDEVBASE	RDEVBLOK	RDEVDED	RDEVDISA	RDEVDISB	RDEVENAB	RDEVFLAG	RDEVLOG
	RDEVSTAT	RDEVTFLG	RDEVTYPC	RDEVTYPE	RDEVUSER	R0	R1	R10	R11	R12	R13	R14	R15
	R2	R3	R4	R5	R6	R7	R8	R9	SAVEAREA	SAVEREGS	SAVER11	SAVER12	SAVEWRK1
	SAVEWRK2	SAVEWRK3	SAVEWRK4	SAVEWRK8	SVMUNLOK	SYSTEM	TIMEDISP	TYPTTY	TYP3066	TYP3277	TYP3278	TYP3284	VCHBLOK
	VCHCUTBL	VCUBLOK	VCUDVTBL	VDEVBLOK	VDEVDED	VDEVFLAG	VDEVSTAT	VDEVTDSK	VDEVTYPC	VMBLOK	VMCHSTRT	VMCHTBL	VMCUSTRT
	VMDSTAT	VMDVSTRT	VMINQ	VMLOGOFF	VMLOGON	VMMACCON	VMMLEVEL	VMPAGES	VMPNT	VMRSTAT	VMSEG	VMSIZE	VMSTOR
	VMUSER	X40FFS	ZEROES										

MODULE	EXTERNAL REFERENCES (LABELS AND MODULES)												
DMKCQG	AFREE	AFRET	AQCNWT	BLANKS	CHXBLOK	CHXOTHR	CHXYADD	CLASDASD	CLASGRAF	CLASSPEC	CLASTAPE	CLASTERM	CLASURI
	CLASURO	DMKCFCSC	DMKCQHPR	DMKCQHPU	DMKCQHRD	DMKCVTBD	DMKCVTBH	DMKCVTHB	DMKERMSG	DMKFREE	DMKFRET	DMKQCNWT	DMKSCNAU
	DMKSCNFD	DMKSCNRD	DMKSCNRN	DMKSCNVN	DMKSCNVU	ERRCODE	F1	F3	F7	F8	NORET	PSA	RANGE
	RDEVBLOK	RDEVSER	RDEVTYPC	RDEVTYPE	R0	R1	R10	R11	R12	R13	R14	R15	R2
	R3	R4	R5	R6	R7	R8	R9	SAVEAREA	SAVEREGS	SAVER0	SAVER11	SAVER2	SAVEWRK1
	SAVEWRK2	SAVEWRK3	SAVEWRK4	SAVEWRK5	SAVEWRK6	SAVEWRK7	SAVEWRK8	TYPBSC	TYPCTCA	TYPPRT	TYPTIMER	TYP2305	TYP2311
	TYP2314	TYP3210	TYP3330	TYP3340	TYP3350	VCHADD	VCHBLOK	VCHCUTEL	VCUADD	VCUBLOK	VCUDVTBL	VDEVADD	VDEVBLOK
	VDEVBND	VDEVCLAS	VDEVCONT	VDEVCOPY	VDEVCSPL	VDEVDED	VDEVENAB	VDEVEOF	VDEVEXTN	VDEVFLAG	VDEVFOR	VDEVHOLD	VDEVNRDY
	VDEVRDO	VDEVREAL	VDEVSFLG	VDEVSTAT	VDEVTDSK	VDEVTERM	VDEVTYPC	VDEVTYPE	VDEVXFER	VDEV231B	VDEV231T	VMBLOK	VMCHSTRT
	VMCHTBL	VMCLASSG	VMCLEVEL	VMCUSTRT	VMDISC	VMDIST	VMDVSTRT	VMFBMX	VMFSTAT	VMOSTAT	VMPSTAT	VMSTKO	VMSTOR
	VMTERM	VMTRMID	VMUSER	VMV370R	VSPXBLOK	VSPXCHAR	VSPXCMCD	VSPXCPYF	VSPXFCB	VSPXFLG1	VSPXFLSH	VSPXOVLY	VSPXXUSR
	ZEROES												
DMKCQH	AFREE	AFRET	AQCNWT	ARSPPR	ARSPPU	ARSPRD	BLANKS	BRING	DMKCVTBD	DMKCVTDB	DMKERMSG	DMKFREE	DMKFRET
	DMKPGTVG	DMKPGTVR	DMKQCNWT	DMKRPAGT	DMKSCNFD	F1	F2	F3	F8	NORET	PSA	RANGE	R0
	R1	R10	R11	R12	R13	R14	R15	R2	R3	R4	R5	R9	SAVEAREA
	SAVEREGS	SAVER0	SAVER2	SAVEWRK1	SAVEWRK2	SAVEWRK5	SAVEWRK7	SAVEWRK8	SFBCLAS	SFBCOPY	SFBDATE	SFBDIST	SFBDUMP
	SFBFILID	SFBFLAG	SFBFLASH	SFBFNAME	SFBFTYPE	SFBINUSE	SFBLOK	SFBORIG	SFBPNT	SFBRECNO	SFBSHOLD	SFBSTART	SFBTIME
	SFBTYPE	SFBUHOLD	SFBUSER	SPCHAR	SPCMOD	SPCOPYFG	SPFCB	SPFLAG1	SPFLSHC	SPLINK	SYSTEM	TYPPRT	TYPRDR
	VMBLOK	VMCLASSD	VMCLEVEL	VMSTKO	VMUSER	ZEROES							
DMKCQP	AFREE	AFRET	ALARM	APSTAT1	APUOPER	AQCNWT	ARIOCH	ARIOCT	ARIOCU	ARIODV	ASYSVM	BLANKS	CLASDASD
	CLASGRAF	CLASSPEC	CLASTAPE	CLASTERM	CLASURI	CLASURO	DMKCFCSC	DMKCVTBD	DMKCVTBH	DMKCVTHB	DMKERMSG	DMKFREE	DMKFRET
	DMKQCNWT	DMKRIORN	DMKRSPUR	DMKSCNAU	DMKSCNFD	DMKSCNRA	DMKSCNRD	DMKSCNRN	DMKSCNRU	DMKSCNVD	DMKSCNVU	DMKSYSRM	DMKSYSRV
	ERRMSG	FFS	F1	F2	F3	F6	F7	IPUADDR	IPUALDRX	NORET	PROCIO	PSA	RANGE
	RCHBLOK	RCHCUTBL	RCUBLOK	RCUCHA	RCUCHAOF	RCUCHBOF	RCUCHCOF	RCUCHD	RCUCHDOF	RCUDISA	RCUDVTBL	RCUPRIME	RCUSTAT
	RCUSUB	RCUTYPE	RDEVACNT	RDEVADD	RDEVATT	RDEVAUTO	RDEVBLOK	RDEVCLAS	RDEVDED	RDEVDISA	RDEVDRAN	RDEVENAB	RDEVFLAG
	RDEVFSEP	RDEVFTR	RDEVIMAG	RDEVLCEP	RDEVLNCP	RDEVLNKS	RDEVMOUT	RDEVNCP	RDEVNRDY	RDEVOVLY	RDEVOWN	RDEVPURG	RDEVSEP
	RDEVSER	RDEVSLOW	RDEVSPL	RDEVSTAT	RDEVSTA2	RDEVSYS	RDEVTYPC	RDEVTYPE	RDEVUSER	RDEVXSEP	RSPLCTL	RSPSFBLK	R0
	R1	R10	R11	R12	R13	R14	R15	R2	R3	R4	R5	R6	R7
	R8	R9	SAVEAREA	SAVEREGS	SAVER0	SAVER1	SAVER11	SAVER2	SAVEWRK1	SAVEWRK2	SAVEWRK3	SAVEWRK4	SAVEWRK5
	SAVEWRK6	SAVEWRK7	SAVEWRK8	SYSVIRT	TEMPR1	TYPBSC	TYP2305	TYP3330	TYP3705	TYP3800	VDEVADD	VDEVBLOK	VDEVBND
	VDEVFLAG	VDEVRDO	VDEVREAL	VDEVRELN	VDEVSIZE	VDEVTDSK	VDEVTYPC	VIRTUAL	VMBLOK	VMDISC	VMDVCNT	VMDVSTRT	VMOSTAT
	VMPNT	VMSTKO	VMTERM	VMTRMID	VMUSER	ZEROES							
DMKCQR	AFREE	AFRET	APSTAT1	APUOPER	AQCNWT	ARIODV	ARSEPR	ARSPPU	ARSPRD	ATTN	BLANKS	CLASDASD	CLASSPEC
	CLASTAPE	CLASTERM	CPMICON	CPSTAT2	DFRET	DMKCVTAB	DMKCVTBD	DMKCVTBH	DMKDMPDV	DMKDMPSW	DMKERMSG	DMKFREE	DMKFRET
	DMKPAGQR	DMKPTRFF	DMKPTRSS	DMKQCNWT	DMKRIOPR	DMKRSPHQ	DMKRSEPR	DMKRSPPU	DMKRSPRD	DMKSCHPG	DMKSCNAU	DMKSCNFD	DMKSCNRD
	DMKSCNRU	F0	F1	F2	F4095	F60	F8	LPUADDR	NICAPL	NICATOF	NICBLOK	NICFLAG	NICLLEN
	NICPSUP	NICSIZE	NICTEXT	NICTMCD	NORET	PAGEWAIT	PREFIXB	PRIORITY	PROCIO	PSA	RDEVAPLP	RDEVATOF	RDEVBLOK
	RDEVFLAG	RDEVLLEN	RDEVNICL	RDEVPSUP	RDEVTEXT	RDEVTFLG	RDEVTMCD	RDEVTYPC	RDEVTYPE	R0	R1	R10	R11
	R12	R13	R14	R15	R2	R3	R4	R5	R6	R7	R8	R9	SAVEAREA
	SAVEREGS	SAVER0	SAVEWRK1	SAVEWRK2	SAVEWRK8	SFBCLAS	SFBFLAG	SFBINUSE	SFBLOK	SFBPNT	SFBSHOLD	SFBUHOLD	SFBUSER
	SHQBLOK	SHQSHOLD	SHQUSER	STARTIME	TEMPSAVE	TYPBSC	TYPPRT	TYPPUN	VMAPF	VMAPFON	VMBLOK	VMCPRUN	VMCLASSD
	VMCLEVEL	VMFAUTO	VMFSTAT	VMGRFTAB	VMISAM	VMMACCON	VMMCODE	VMMCPENV	VMMCR6	VMMFE	VMMIMSG	VMMLEVEL	VMMLINED
	VMMLVL2	VMMSGON	VMMSVC	VMMTEXT	VMMVTMR	VMNOTRAN	VMOSTAT	VMPAGEX	VMPNT	VMPSTAT	VMRON	VMSPMFLG	VMSPMON

MODULE	EXTERNAL REFERENCES (LABELS AND MODULES)												
	VMSTKO	VMTCDEL	VMTERM	VMTESCP	VMTLDEL	VMTLEND	VMTLEVEL	VMTON	VMTRMID	VMUPRIOR	VMUSER	VMV370R	VMWNGON
	ZEROES												
DMKCQY	AFREE	AFRET	APSTAT1	APUOPER	AQCNWT	BLANKS	CLASSPEC	CLASTERM	CPASTCN	CPMICON	CPSTAT2	CPUID	DFRET
	DMKACOTM	DMKCVTBD	DMKCVTBH	DMKCVTDB	DMKCVTDT	DMKERMSG	DMKFREE	DMKFRET	DMKQCNWT	DMKSCNAU	DMKSCNFD	DMKSCNRD	DMKSYSDW
	DMKSYSLG	DMKSYSND	DMKSYSNM	DMKSYSTI	F1	F2	F4	F8	IPUADDR	IPUADDRX	IS	NORET	PREFIXB
	PSA	RDEVBLOK	RDEVTYPC	RDEVTYPE	R0	R1	R10	R11	R12	R13	R14	R15	R2
	R3	R4	R5	R6	R7	R8	R9	SAVEAREA	SAVEREGS	SAVER0	SAVEWRK1	SAVEWRK2	SAVEWRK4
	SAVEWRK5	SAVEWRK6	SAVEWRK7	SAVEWRK9	TYPBSC	VMBLOK	VMCPUID	VMDISC	VMOSTAT	VMPFUNC	VMPNT	VMSTKO	VMTERM
	VMTLEND	VMTRMID	VMUSER	ZEROES									
DMKCSB	ACORETBL	ADSPCH	AFREE	AFRET	APTRAN	AQCNWT	ARIOPR	ARIOPU	ARIORD	ASYSVM	BLANKS	BRING	BUFFER
	CC	CLASURI	CLASURO	C1	DEFER	DMKCVTDB	DMKCVTHB	DMKDSPCH	DMKERMSG	DMKFCBLD	DMKFREE	DMKFRET	DMKIOSQR
	DMKPTRUL	DMKQCNWT	DMKSCNFD	DMKSCNRU	DMKSCNVU	DMKUCBLD	DMKUCCLD	DMKUCSLD	FCBNEXT	FTRUCS	F1	F2	F3
	F4	IOBCAW	IOBCSW	IOBFATAL	IOBFLAG	IOBIRA	IOBLINK	IOBLOK	IOBMISC2	IOBRADD	IOBRSTRT	IOBSIZE	IOBSTAT
	IOBUSER	LOCK	NORET	PSA	RDEVBLOK	RDEVDED	RDEVDISA	RDEVDRAN	RDEVFLAG	RDEVFTR	RDEVSPL	RDEVSTAT	RDEVTYPC
	RDEVTYPE	RDEVUSER	R0	R1	R10	R11	R12	R13	R14	R15	R2	R3	R4
	R5	R6	R8	R9	SAVEAREA	SAVEREGS	SAVEWRK1	SAVEWRK2	SAVEWRK4	SAVEWRK6	SAVEWRK7	SAVEWRK8	SILI
	SKIP	SYSTEM	TYPPRT	TYP3203	TYP3211	UE	VDEVBLOK	VDEVDED	VDEVFCBK	VDEVSTAT	VDEVTYPE	VFCBBLOK	VFCBCNT
	VFCBLOAD	VFCBNDEX	VFCBSIZE	VMBLOK	VMSEG	VMUSER	ZEROES						
DMKCSO	AFREE	AFRET	APSTAT1	APTRAN	AQCNWT	ARIODV	ARIOPR	ARIOPU	ARIORD	ASYSVM	BALRSAVE	BLANKS	BRING
	BUFFER	BUFNXT	CHGRDV	CLASURI	CLASURO	C1	DE	DEFER	DEVADDR	DEVTYPE	DMKCKSPL	DMKCVTBH	DMKCVTDB
	DMKCVTHB	DMKERMSG	DMKFREE	DMKFRET	DMKPTRAN	DMKQCNWT	DMKQNTBL	DMKRSPEX	DMKSCNFD	DMKSCNRU	DMKSPLDL	DMKSTKIO	F1
	F2	F3	F4	F6	F8	IOBCP	IOBCSW	IOBFATAL	IOBFLAG	IOBIRA	IOBLINK	IOBLOK	IOBMISC
	IOBRADD	IOBSIZE	IOBSTAT	IOBUSER	LOCK	NORET	NPRCNT	NPRNAME	NPRPNT	NPRTBL	OPERATOR	PROCIO	PSA
	RDEVACNT	RDEVAIOB	RDEVBACK	RDEVBLOK	RDEVBUSY	RDEVCLAS	RDEVDED	RDEVDISA	RDEVDRAN	RDEVFLAG	RDEVFSEP	RDEVIMAG	RDEVIOER
	RDEVLOAD	RDEVNRDY	RDEVOVLY	RDEVPURG	RDEVRSTR	RDEVSEP	RDEVSPAC	RDEVSPL	RDEVSTAT	RDEVSTA2	RDEVTERM	RDEVTYPC	RDEVTYPE
	RDEVUSER	RDEVXSEP	RSPLCTL	RSPMISC	RSPSFBLK	R0	R1	R10	R11	R12	R13	R14	R15
	R2	R3	R4	R5	R6	R7	R8	R9	SAVEAREA	SAVEREGS	SAVEWRK1	SAVEWRK2	SAVEWRK3
	SAVEWRK4	SAVEWRK5	SAVEWRK6	SAVEWRK7	SAVEWRK8	SFBCOPY	SFBFLAG	SFBFLAG2	SFBLCK	SFBRECER	SFBRECOK	SFBREQUE	SFBSHOLD
	SYSTEM	TYPPRT	TYPPUN	TYP3800	VMBLOK	VMOSTAT	VMSEG	VMSYSOP	VMUSER	ZEROES			
DMKCSP	AFREE	AFRET	ARSPPR	ARSPPU	ARSPRD	BLANKS	BUFFER	CLASTERM	CLASURI	CLASURO	DE	DMKCVTDB	DMKCVTHB
	DMKERMSG	DMKFREE	DMKFRET	DMKSCNFD	DMKSCNVU	DMKSTKIO	DMKUDRFU	DMKVIOIN	DMKVSPCO	FFS	F1	F2	F3
	F4	F8	IOBCSW	IOBIRA	IOBLINK	IOBLOK	IOBSIZE	IOBUSER	IOBVADD	LOCK	PSA	R0	R1
	R10	R11	R12	R13	R14	R15	R2	R3	R4	R5	R6	R7	R8
	R9	SAVEAREA	SAVEREGS	SAVEWRK1	SAVEWRK2	SAVEWRK4	SAVEWRK5	SAVEWRK6	SAVEWRK7	SAVEWRK8	SAVEWRK9	SFBCLAS	SFBFILID
	SFBFLAG	SFBINUSE	SFBLOK	SFBUHOLD	SFBUSER	TEMPR2	TYPPRT	TYPPUN	TYPRDR	TYP3210	UDIRBLOK	UDIRPASS	UDIRSIZE
	VCHADD	VCHBLOK	VCHCUTBL	VCUADD	VCUBLOK	VCUDVTBL	VDEVADD	VDEVBLOK	VDEVCLAS	VDEVCONT	VDEVCOPY	VDEVCSPL	VDEVCSW
	VDEVDED	VDEVEOF	VDEVEXTN	VDEVFLAG	VDEVFOR	VDEVHOLD	VDEVPEND	VDEVPURG	VDEVSPLG	VDEVSIZE	VDEVSPL	VDEVSTAT	VDEVTERM
	VDEVTYPC	VDEVTYPE	VDEVXFER	VMBLOK	VMCHSTRT	VMCHTBL	VMCUSTRT	VMDVCNT	VMDVSTRT	VMUSER	VSPLCTL	VSPSFBLK	VSPXBLOK
	VSPXCHAR	VSPXCMOD	VSPXCPYF	VSPXFCB	VSPXFLG1	VSPXFLSH	VSPXOVLY	VSPXXUSR	ZEROES				
DMKCSQ	AFREE	AFRET	ARSPPR	ARSPPU	ARSPRD	BLANKS	BUFFER	CHGSFB	CHGSHQ	CLASTERM	CLASURI	CLASURO	DELSFB
	DMKCKSPL	DMKCSOSD	DMKCVTHB	DMKERMSG	DMKFREE	DMKFRET	DMKRSPHQ	DMKSCNFD	DMKSCNVU	DMKUDRFU	DMKVSPCO	DMKVSPCR	FFS

MODULE	EXTERNAL REFERENCES (LABELS AND MODULES)												
	F1	F2	F3	F7	F8	LOCK	PSA	R0	R1	R10	R11	R12	R13
	R14	R15	R2	R3	R4	R5	R6	R7	R8	R9	SAVEAREA	SAVEREGS	SAVEWRK1
	SAVEWRK2	SAVEWRK4	SAVEWRK5	SAVEWRK6	SAVEWRK8	SAVEWRK9	SFBCLAS	SFBDIST	SFBFILID	SFBFLAG	SFBFLAG2	SFBFNAME	SFBHOLD
	SFBINUSE	SFBLOK	SFBNOHLD	SFBSHOLD	SFBUSER	SHQBLOK	SHQBSIZE	SHQFLAGS	SHQPNT	SHQSHOLD	SHQUSER	TYPPRT	TYPPUN
	TYPRDR	TYP3210	VDEVADD	VDEVBLOK	VDEVCONT	VDEVDED	VDEVPURG	VDEVSFLG	VDEVSIZE	VDEVSPL	VDEVSTAT	VDEVTYPC	VDEVTYPE
	VMBLOK	VMDVCNT	VMDVSTRT	VSPLCTL	VSPSFBLK	ZEROES							
DMKCST	AFREE	AFRET	ARSPRD	BRING	BUFCNT	BUFFER	BUFNXT	CLASTERM	CLASURI	CLASURO	DMKCVTBH	DMKCVTDB	DMKCVTHB
	DMKERMSG	DMKFREE	DMKFRET	DMKPGTVG	DMKPGTVR	DMKRPAGT	DMKRPAPT	DMKSCNFD	DMKSCNVD	DMKSCNVN	DMKSCNVU	FFS	F1
	F2	F3	LOCK	PSA	R0	R1	R10	R11	R12	R13	R14	R15	R2
	R3	R4	R5	R6	R7	R8	R9	SAVEAREA	SAVEREGS	SAVEWRK1	SAVEWRK2	SAVEWRK4	SAVEWRK5
	SAVEWRK6	SAVEWRK7	SAVEWRK8	SFBFILID	SFBLOK	SFBPNT	SFBSTART	SFBUSER	SKIP	SYSTEM	TAG	TYPPRT	TYPPUN
	TYPRDR	TYP3210	VDEVADD	VDEVBLOK	VDEVDED	VDEVEXTN	VDEVSIZE	VDEVSTAT	VDEVTYPC	VDEVTYPE	VMBLOK	VMDVCNT	VMDVSTRT
	VMSTKO	VMUSER	VSPXBLOK	VSPXTAG	VSPXTGLN	ZEROES							
DMKCSU	AFREE	AFRET	AQCNWT	ARSPPR	ARSPPU	ARSPRD	BLANKS	BRING	BUFFER	BUFNXT	CHGSFB	CLASURI	COUNT
	DE	DMKCKSPL	DMKCSOSD	DMKCVTBD	DMKCVTDB	DMKERMSG	DMKFREE	DMKFRET	DMKPGTVG	DMKPGTVR	DMKQCNWT	DMKRPAGT	DMKRPAPT
	DMKSCNAU	DMKSCNFD	DMKSTKIO	DMKUDRFU	DMKVIOIN	FFS	F1	F2	F24	F3	F4	F5	F6
	F7	F8	IOBCSW	IOBIRA	IOBLINK	IOBLOK	IOBSIZE	IOBUSER	IOBVADD	LOCK	NORET	PSA	R0
	R1	R10	R11	R12	R13	R14	R15	R2	R3	R4	R5	R6	R7
	R8	R9	SAVEAREA	SAVEREGS	SAVEWRK1	SAVEWRK2	SAVEWRK4	SAVEWRK5	SAVEWRK6	SAVEWRK7	SAVEWRK8	SFBCLAS	SFBCOPY
	SFBDIST	SFBFILID	SFBFLAG	SFBFLASH	SFBFNAME	SFBINUSE	SFBLOK	SFBSHOLD	SFBSTART	SFBUHOLD	SFBUSER	SPCHAR	SPCMOD
	SPCOPYFG	SPFCB	SPFLAG1	SPFLSHC	SPLINK	SYSTEM	TEMPR2	TEMPR3	TEMPR4	TIMEDISP	TYPPRT	TYPPUN	TYPRDR
	VCHADD	VCHBLOK	VCHCUTBL	VCUADD	VCUBLOK	VCUDVTBL	VDEVADD	VDEVBLOK	VDEVCLAS	VDEVCSW	VDEVPEND	VDEVSPL	VDEVSTAT
	VDEVTYPC	VDEVTYPE	VMBLOK	VMCHSTRT	VMCHTBL	VMCLASSD	VMCLEVEL	VMCOMND	VMCUSTRT	VMDVSTRT	VMMIMSG	VMMLVL2	VMUSER
	X2048BND	ZEROES											
DMKCSV	AFREE	AFRET	APSTAT1	APUOPER	AQCNWT	ARSPPR	ARSPPU	ARSPRD	BLANKS	BUFFER	BUFNXT	CHGSFB	CLASURI
	COUNT	DE	DMKCKSPL	DMKCVTBD	DMKCVTDB	DMKERMSG	DMKFREE	DMKFRET	DMKLOKSW	DMKQCNWT	DMKSCNAU	DMKSCNFD	DMKSPLDL
	DMKSTKIO	DMKUDRFU	DMKVIOIN	FFS	F1	F2	F3	F4	F6	F8	IOBCSW	IOBIRA	IOBLINK
	IOBLOK	IOBSIZE	IOBUSER	IOBVADD	LOCK	NORET	NOTRESP	PSA	R0	R1	R10	R11	R12
	R13	R14	R15	R2	R3	R4	R5	R6	R7	R8	R9	SAVEAREA	SAVEREGS
	SAVER11	SAVEWRK1	SAVEWRK2	SAVEWRK4	SAVEWRK5	SAVEWRK6	SAVEWRK7	SAVEWRK8	SAVEWRK9	SFBCLAS	SFBFILID	SFBFLAG	SFBINUSE
	SFBLOK	SFBORIG	SFBPNT	SFBUSER	TEMPR2	TEMPR3	TEMPR4	TIMEDISP	TYPPRT	TYPPUN	TYPRDR	VCHADD	VCHBLOK
	VCHCUTBL	VCUADD	VCUBLOK	VCUDVTEL	VDEVADD	VDEVBLOK	VDEVCLAS	VDEVCSW	VDEVPEND	VDEVSPL	VDEVSTAT	VDEVTYPC	VDEVTYPE
	VMBLOK	VMCHSTRT	VMCHTBL	VMCLASSD	VMCLEVEL	VMCOMND	VMCUSTRT	VMDVSTRT	VMMIMSG	VMMLEVEL	VMMLVL2	VMMSGON	VMUSER
	ZEROES												
DMKCVT	BALRSAVE	BALR1	BALR2	CPID	DATE	F1	F10	F240	F4	F60	LOCK	PACK	PSA
	R0	R1	R10	R14	R15	R2	R3	R5	R6	R7	R8	R9	TEMPSAVE
	TODATE												
DMKDAS	AFREE	AFRET	ALARM	APSTAT1	AQCNWT	ASYSOP	BLANKS	CC	CCC	CCW1	CCW2	CCW3	CD
	CDC	CL	DFRET	DMKCVTBH	DMKFREE	DMKFRET	DMKIOEST	DMKMSWR	DMKQCNWT	DMKTRKIN	FFS	FTREXTSN	FTRRPS
	FTRRSRL	FTR35MB	FTR70MB	F1	F10	F16	F2	F256	F4095	F4096	F8	IDA	IFCC
	IOBALTSK	IOBCAW	IOBCP	IOBCSW	IOBERP	IOBFATAL	IOBFLAG	IOBIOER	IOBLOK	IOBRADD	IOBRCAW	IOBRCNT	IOBREL

MODULE	EXTERNAL REFERENCES (LABELS AND MODULES)												
	IOBRSTRT	IOBSPEC2	IOBSTAT	IOERACT	IOERADR	IOERALTR	IOERBLCK	IOERCAL	IOERCAN	IOERCCRA	IOERCCRL	IOERCEMD	IOERCSW
	IOERDASD	IOERDATA	IOERDEC	IOERDW	IOERECF	IOERETRY	IOEREXT	IOERFLG1	IOERFLG2	IOERFLG3	IOERHA	IOERIGNR	IOERIND3
	IOERIND4	IOERINFO	IOERLOC	IOERMSG	IOERMSW	IOERNUM	IOERPEND	IOERPNT	IOERRDR0	IOERREAD	IOERSIZE	IOERSNSZ	IOERSTAT
	IOERSTRT	IOERVOL1	IOERVSER	LOCK	NORET	OPERATOR	PRGC	PROCIO	PRTC	PSA	RDCOUNT	RDEVBLOK	RDEVCUB
	RDEVDED	RDEVDISA	RDEVFLAG	RDEVFTR	RDEVIOER	RDEVMOUT	RDEVNRDY	RDEVOWN	RDEVSER	RDEVSTAT	RDEVSYS	RDEVTYPE	R0
	R1	R10	R11	R12	R13	R14	R15	R2	R3	R4	R5	R6	R7
	R8	R9	SAVEAREA	SAVEREGS	SAVER11	SAVEWRK2	SAVEWRK3	SEEK	SEEKADR	SILI	SKIP	TYP2305	TYP2314
	TYP3330	TYP3340	TYP3350	UC	VDEVBLOK	VDEVCSPL	VDEVFLAG	VMBLOK	VMDVSTRT	VMVTERM	XRIGHT16	ZEROES	
DMKDDR	ATTN	BLANKS	BUSY	CAW	CC	CD	CE	CLASDASD	CLASTAPE	CLASTERM	CONPARM	CPUID	CSW
	CUE	DATE	DE	ERRMSG	ERROR	ERSAVE	EUA	EXTSIZE	IC	INPUT	INTREQ	IOBCSW	IOBSIZE
	IOBSTAT	IOOLD	NEWEXT	OUTPUT	PACK	PRINTER1	PRINTER2	RA	R0	R1	R10	R11	R12
	R13	R14	R15	R2	R3	R4	R5	R6	R7	R8	R9	SAVEAREA	SAVEREGS
	SBA	SENSE	SF	SILI	SKIP	TAPE	TESTCON	TEXTA	TIMER	TODATE	TYPE	TYP2305	TYP2311
	TYP2314	TYP2319	TYP2401	TYP2415	TYP2420	TYP3330	TYP3340	TYP3350	TYP3410	TYP3411	TYP3420	UC	UE
	WAIT												
DMKDEF	AFREE	AFRET	AQCNWT	BLANKS	CLASDASD	CLASGRAF	CLASSPEC	CLASTERM	CLASURI	CLASURO	DELPAGES	DELSEGS	DMKBLDRL
	DMKBLDRT	DMKCFPRD	DMKCFPRR	DMKCVTBD	DMKCVTBH	DMKCVTDB	DMKCVTHB	DMKERMSG	DMKFREE	DMKFRET	DMKLOCKD	DMKLOCKQ	DMKPGSPO
	DMKQCNWT	DMKSCNFD	DMKSCNRU	DMKSCNVD	DMKSCNVN	DMKSCNVU	DMKSSSDE	DMKUDRFU	DMKUDRMD	DMKUDRRV	DMKVCARS	DMKVDSDF	FFS
	F16	F2	F3	F4	F5	F8	LOCK	MSSPRES	NEWPAGES	NEWSEGS	NORET	PSA	PSAMSS
	RDEVADD	RDEVATT	RDEVBLOK	RDEVDED	RDEVDISA	RDEVFLAG	RDEVFTR	RDEVLNKS	RDEVSER	RDEVSTAT	RDEVTYPC	R0	R1
	R10	R11	R12	R13	R14	R15	R2	R3	R4	R5	R6	R7	R8
	R9	SAVEAREA	SAVEREGS	SAVER2	SAVEWRK1	SAVEWRK2	SAVEWRK3	SAVEWRK4	SAVEWRK5	SAVEWRK6	SAVEWRK7	SAVEWRK8	SAVEWRK9
	SYSVIRT	TYPCTCA	TYPIBM1	TYPPRT	TYPTELE2	TYP1052	TYP1403	TYP2305	TYP3203	TYP3211	UDBFBLOK	UDBFSIZE	UDBFVADD
	UDEVADD	UDEVBLOK	UDEVCLAS	UDEVDISP	UDEVFTR	UDEVNCYL	UDEVSTAT	UDEVTDSK	UDEVTYPC	UDEVTYPE	UDEV3158	UDIRBLOK	UDIRDISP
	UMACBLOK	UMACMCOR	VCHADD	VCHBLOK	VCHBMX	VCHCUTBL	VCHDED	VCHSEL	VCHSTAT	VCHTYPE	VCUADD	VCUBLOK	VCUCTCA
	VCUDVTBL	VCUTYPE	VDEVADD	VDEVBLOK	VDEVDED	VDEVFLAG	VDEVFLG2	VDEVLINK	VDEVPOSN	VDEVRELN	VDEVSTAT	VDEVTDSK	VDEVTYPC
	VIRTUAL	VMBLOK	VMCHSTRT	VMCHTBL	VMCLASSA	VMCLASSB	VMCLEVEL	VMCUSTRT	VMDVSTRT	VMFBMX	VMFSTAT	VMINHMIG	VMMIMSG
	VMMLVL2	VMPSTAT	VMQSTAT	VMREAL	VMSIZE	VMSTOR	VMUSER	VMVTERM	VMV370R	VRALOC			
DMKDGD	ACORETBL	ADSPCH	AFREE	AFRET	APTRAN	APTRLK	BRING	CC	CD	CLASDASD	CORFLAG	CORFLUSH	CORPGPNT
	CORSWPNT	CORTABLE	CPEXADD	CPEXBLOK	CPEXFPNT	CPEXMISC	CPEXREGS	CPEXR0	CPEXSIZE	CPSHRLK	CPSTAT2	CSW	C1
	DEFER	DMKDSPCH	DMKFREE	DMKFRET	DMKIOSQV	DMKPSACC	DMKPSASC	DMKPTRAN	DMKPTRFR	DMKPTRFT	DMKPTRUL	DMKSCNVU	DMKSSSMQ
	DMKVMASH	FFS	F1	F15	F16	F3	F4	F4095	F4096	F5	F6	F8	IDA
	IOBCAW	IOBCC1	IOBCC3	IOBCSW	IOBCYL	IOBFATAL	IOBFLAG	IOBHVC	IOBIOER	IOBIRA	IOBLINK	IOBLOK	IOBMISC
	IOBMISC2	IOBSIZE	IOBSTAT	IOERBLOK	IOEREXT	IOERSIZE	LOCK	MSSNEXT	MSSPRES	MSSTASK1	MSSTASK2	MSSUSER	PCIF
	PSA	PSAMSS	RCWADDR	RCWCCW	RCWCNT	RCWCOMND	RCWCTL	RCWFLAG	RCWIO	RCWSHR	RDEVBLOK	RDEVFTR	R0
	R1	R10	R11	R12	R13	R14	R15	R2	R3	R4	R5	R6	R7
	R8	R9	SAVEAREA	SAVEREGS	SAVER12	SAVER2	SAVEWRK9	SHRLKCNT	SILI	SWPFLAG	SYSVIRT	TYP2305	TYP2311
	TYP2314	TYP3330	TYP3340	TYP3350	VDEVBLOK	VDEVBND	VDEVBUSY	VDEVCHAN	VDEVCPEX	VDEVDED	VDEVFLAG	VDEVFLG2	VDEVIOB
	VDEVIOER	VDEVPEND	VDEVPOSN	VDEVRDO	VDEVREAL	VDEVRELN	VDEVRES	VDEVRRB	VDEVRRF	VDEVSTAT	VDEVTYPC	VDEVTYPE	VDEVUC
	VIRTUAL	VMACTDEV	VMBLOK	VMCFWAIT	VMCOMP	VMDVSTRT	VMESTAT	VMEXTCM	VMEXWAIT	VMGPRS	VMIDLE	VMIOCNT	VMIOWAIT
	VMLOGOFF	VMLOPRI	VMPSW	VMQLEVEL	VMRSTAT	VMSEG	VMUSER	VRRBLOK	VRRCPEX	VRRRES	VRRSTAT	XPAGNUM	
DMKDIA	ADSPCH	AFREE	AFRET	APSTAT1	APUOPER	AQCNWT	ARIOCU	ARIODV	ASYSVM	BALRSAVE	BALR1	BLANKS	CCDESMD

MODULE	EXTERNAL REFERENCES (LABELS AND MODULES)												
	CHBSIZE	CHXBLOK	CHXOTHR	CHXYADD	CHYBLOK	CHYOTHR	CHYXADD	CLASGRAF	CLASSPEC	CLASTERM	CONCCW3	CONDATA	CONDCNT
	CONSYSR	CPEXADD	CPEXBLOK	CPEXSIZE	CRESDQ	CRESIMD	CSETDSM	CSWLMEP	CSWLNCP	CTRMLTR	DE	DFRET	DMKACODV
	DMKBLDVM	DMKCFPRD	DMKCVTAB	DMKCVTBD	DMKCVTBH	DMKCVTHB	DMKDSPCH	DMKERMSG	DMKFREE	DMKFRET	DMKIOSHA	DMKIOSQR	DMKLOKSW
	DMKQCNCL	DMKQCNWT	DMKRIORN	DMKRNHND	DMKSCHRT	DMKSCNAU	DMKSCNFD	DMKSCNRD	DMKSCNRN	DMKSCNRU	DMKSCNVD	DMKSCNVU	DMKSTKCP
	DMKSTKIO	DMKSYSCK	DMKSYSND	DMKVCARS	DMKVIOIN	FFS	F1	F240	F3	F4095	GRAFDEV	IOBCP	IOBCSW
	IOBFLAG	IOBIOER	IOBIRA	IOBLINK	IOBLOK	IOBMISC	IOBRADD	IOBRCAW	IOBSIZE	IOBUSER	IOBVADD	IOERBLOK	IOEREXT
	IOERSIZE	LASTUSER	LOCK	LOGHOLD	NICBLOK	NICCIBM	NICDISA	NICENAB	NICEPAD	NICEPMD	NICFLAG	NICLINE	NICLTRC
	NICNAME	NICQPNT	NICSESN	NICSIZE	NICSTAT	NICSWEP	NICTELE	NICTYPE	NICUSER	NORET	OPERATOR	PRIORITY	PROCIO
	PSA	RCHBLOK	RCHCUTBL	RCUBLOK	RCUDVTBL	RDEVACTV	RDEVADD	RDEVAIOB	RDEVAIRA	RDEVATT	RDEVBASE	RDEVBLOK	RDEVCON
	RDEVCORD	RDEVCTL	RDEVCUA	RDEVCYL	RDEVDED	RDEVEPDV	RDEVEPLN	RDEVEPMD	RDEVFLAG	RDEVHIO	RDEVLCEP	RDEVLNCP	RDEVNICL
	RDEVNRDY	RDEVPREP	RDEVRCVY	RDEVRUN	RDEVSTAT	RDEVTFLG	RDEVTMAT	RDEVTYPC	RDEVTYPE	RDEVUSER	RUNUSER	R0	R1
	R10	R11	R12	R13	R14	R15	R2	R3	R4	R5	R6	R7	R8
	R9	SAVEAREA	SAVEREGS	SAVERETN	SAVER11	SAVER2	SAVER8	SAVEWRK1	SAVEWRK2	SAVEWRK3	SAVEWRK4	SAVEWRK5	SAVEWRK6
	SAVEWRK7	SAVEWRK8	SAVEWRK9	START	TIMEDISP	TRQBFPNT	TRQBLCK	TRQBSIZE	TYPESC	TYPCTCA	TYPIBM1	TYPTELE2	TYP3210
	TYP3277	VCHADD	VCHBLOK	VCHCUTEL	VCUADD	VCUBLOK	VCUDVTBL	VDEVADD	VDEVBLOK	VDEVDED	VDEVDIAL	VDEVENAB	VDEVFLAG
	VDEVIOB	VDEVNRDY	VDEVREAL	VDEVSTAT	VDEVTYPC	VDEVTYPE	VMBLOK	VMBSIZE	VMCF	VMCHSTRT	VMCHTBL	VMCUSTRT	VMDELAY
	VMDFTPNT	VMDVSTRT	VMKILL	VMLOGOFF	VMOSTAT	VMPNT	VMPSTAT	VMRSTAT	VMTERM	VMTRMID	VMUSER	VMV370R	ZEROES
DMKDIB	AFREE	CC	CD	CE	CMDREJ	DE	DMKDIADR	DMKFREE	DMKSTKIO	DMKSYSRM	F240	IDA	IL
	INTREQ	IOBCAW	IOBCC1	IOBCSW	IOBFLAG	IOBIOER	IOBLOK	IOBRCAW	IOBRSTRT	IOBSTAT	IOBUSER	IOERBLOK	IOERCCW
	IOERCSW	IOERDATA	IOERSIZE	LOCK	PRGC	PRTC	PSA	RCWADDR	RCWCCW	RCWCNT	RCWCOMND	RCWCTL	RCWFLAG
	RCWINVL	R0	R1	R10	R11	R12	R13	R14	R15	R2	R4	R6	R7
	R8	R9	SAVEAREA	SAVEREGS	SAVER2	SAVEWRK1	SILI	SKIP	TYPCTCA	TYP3277	UC	VDEVBLOK	VDEVDIAL
	VDEVENAB	VDEVFLAG	VDEVIOB	VMBLOK	VMDVSTRT	VMIOWAIT	VMRSTAT	ZEROES					
DMKDIR	ATTN	BUSY	CAW	CC	CD	CE	CLASDASD	CLASGRAF	CLASSPEC	CLASTERM	CLASURI	CLASURO	CONCCW1
	CONPARM	CONSOLE	CONUSER	CPUID	CSW	CUE	DE	DIRPTR	ERRMSG	ERROR	EUA	FLAG	FTR2311B
	FTR2311T	F4096	HEX	IC	IOBCSW	IOBSIZE	IOBSTAT	ICOLD	PACK	POINTERS	RA	READ	R0
	R1	R10	R11	R12	R13	R14	R15	R2	R3	R4	R5	R6	R7
	R9	SAVEAREA	SAVEREGS	SBA	SENSE	SF	SILI	SKIP	TEXTA	TIMER	TYPCTCA	TYPE	TYPIBM1
	TYPTELE2	TYPTIMER	TYP1052	TYP1403	TYP1443	TYP2305	TYP2311	TYP2314	TYP2501	TYP2540P	TYP2540R	TYP3138	TYP3148
	TYP3158	TYP3203	TYP3210	TYP3211	TYP3215	TYP3277	TYP3330	TYP3340	TYP3350	TYP3505	TYP3525	UC	UDEVADD
	UDEVBLOK	UDEVCLAS	UDEVDASD	UDEVDED	UDEVDISP	UDEVFTR	UDEVLINK	UDEVLKDV	UDEVLKID	UDEVLM	UDEVLONG	UDEVLR	UDEVLW
	UDEVMODE	UDEVMR	UDEVMW	UDEVNCYL	UDEVPASM	UDEVPASR	UDEVPASW	UDEVR	UDEVRELN	UDEVRR	UDEVSIZE	UDEVSPOO	UDEVSTAT
	UDEVTDSK	UDEVTYPC	UDEVTYPE	UDEVVRR	UDEVVSER	UDEVW	UDEVWR	UDEV3158	UDIRBLOK	UDIRDASD	UDIRDISP	UDIRPASS	UDIRSIZE
	UDIRUSER	UE	UMACACC	UMACACCT	UMACAFF	UMACBLOK	UMACBMX	UMACCDEL	UMACCLEV	UMACCORE	UMACCPU	UMACDASD	UMACDISP
	UMACDIST	UMACDVCT	UMACECOP	UMACES	UMACFFON	UMACIPL	UMACISAM	UMACLDEL	UMACLEND	UMACMCOR	UMACNSVC	UMACOPT	UMACOPT2
	UMACPRIR	UMACPUID	UMACRT	UMACSIZE	UMACVROP	VIRTUAL	WRITE						
DMKDMP	ALARM	ATTN	BALR2	BUSY	CAW	CC	CE	CHGSFB	CLASDASD	CLASTAPE	CLASURO	CORCP	CORFLAG
	CORFPNT	CORTABLE	CPABEND	CPID	CPULOG	CSW	CUE	C0	C14	C15	C2	DAMAGRPT	DE
	DMKOPRWT	DMKPRGMC	DMKRIODV	DMKRIOPR	DMKRSPID	DMKRSPRD	DMKSCNRD	DMKSCNRU	DMKSYSCH	DMKSYSCK	DMKSYSCS	DMKSYSRM	DMKSYSRV
	DMKSYSVM	DMPABEND	DMPFLAG	DMPFPRS	DMPGPRS	DMPINREC	DMPKEY	DMPKYREC	DMPLCORE	DMPPGMAP	DMPPRFRG	DMPPROCA	DMPSYSRV
	DMPTODCK	DUMPSAVE	EXTMODE	FPRLOG	FXDLOG	GRLOG	HALFPAGE	HARDSTOP	INTPRL	INTREQ	INTTIO	IOBSIZE	IOMASK
	IONPSW	IPLCCW1	IPLPSW	KEY	LASTLINE	LINE	LOCK	MCHEK	MONAIOB	MONARDB	MONCOM	MONFLAG2	MONFLAG3
	PRNPSW	PROPSW	PSA	RDEVBLOK	RDEVRECS	RDEVTYPC	RDEVTYPE	RDRCHN	RECBLOK	RECCYL	RECMAP	RECPNT	RECUSED

MODULE	EXTERNAL REFERENCES (LABELS AND MODULES)												
	RSRTNPSW	R0	R1	R10	R11	R12	R13	R14	R15	R2	R3	R4	R5
	R6	R7	R8	R9	SEEK	SENSE	SFBDATE	SFBDUMP	SFBFILID	SFBLAST	SFBLOK	SFBPNT	SFBSIZE
	SFBSTART	SFBTIME	SFBTYPE	SIGREST	SIGSENSE	SIGSSS	SIGSTOP	SILI	SKIP	SM	SPOOLED	TODATE	TRUN
	TYPPRT	TYP1403	TYP2314	TYP3330	TYP3340	TYP3350	TYP3420	UC	UE	WAIT	Y0	Y2	Y4
	Y6												
DMKDRD	AFREE	AFRET	APTRAN	ARIODV	ARSPRD	ASYSVM	BRING	CLASURI	C1	DEFER	DMKFREE	DMKFRET	DMKHVCPC
	DMKPGTSD	DMKPGTVG	DMKPGTVR	DMKPSASP	DMKPTRAN	DMKRPAGT	DMKSCNVU	DMKSYM	DMKSYSCW	DMKVSPCR	FFS	F1	F255
	F256	F4096	F8	LOCK	OWNDLIST	OWNDRDEV	PSA	RDEVBLOK	RDEVTYPE	R0	R1	R10	R11
	R12	R13	R14	R15	R2	R3	R4	R5	R6	R7	R8	R9	SAVEAREA
	SAVEREGS	SAVER0	SAVER2	SAVER6	SAVEWRK1	SAVEWRK2	SAVEWRK3	SAVEWRK4	SAVEWRK5	SAVEWRK6	SFBCLAS	SFBCOPY	SFBDUMP
	SFBEOF	SFBFILID	SFBFLAG	SFBFLAG2	SFBINUSE	SFBLAST	SFBLOK	SFBMON	SFBOPEN	SFBPNT	SFBRECER	SFBSIZE	SFBSTART
	SFBTYPE	SFBUHOLD	SFBUSER	SKIP	SPLINK	SPNXTPAG	SPPREPAG	SYSTEM	TYPPRT	TYPPUN	TYPRDR	TYP2305	TYP2319
	TYP3330	TYP3340	TYP3350	VDEVBLOK	VDEVBUSY	VDEVCLAS	VDEVCCNT	VDEVDIAG	VDEVSFLG	VDEVSPL	VDEVSTAT	VDEVTYPC	VDEVTYPE
	VMBLOK	VMDVSTRT	VMESTAT	VMEXTCM	VMPSTAT	VMPSW	VMSEG	VMUSER	VMV370R	VSPCAW	VSPCCW	VSPDPAGE	VSPLCTL
	VSPSFBLK	VSPSIZE	XPAGNUM	X2048BND	ZEROES								
DMKDSB	ADSPCH	AFREE	AFRET	ALARM	APSTAT1	AQCNWT	ASYSVM	BLANKS	CC	CCC	CDC	CL	CPEXADD
	CPEXBLOK	CPEXREGS	CPEXSIZE	DE	DFRET	DMKCVTBH	DMKDSPCH	DMKFREE	DMKFRET	DMKIOESD	DMKIOSQR	DMKQCNWT	DMKSCNRU
	DMKSSSEN	DMKSSSMQ	DMKSTKCP	FTREXTSN	FIRRPS	FTRRSRL	FTR35MB	FTR70MB	IFCC	IOBCAW	IOBCC3	IOBCSW	IOBFATAL
	IOBIOER	IOBIRA	IOBLINK	IOBLOK	IOBRADD	IOBREL	IOBSIZE	IOBSPEC	IOBSPEC2	IOBSTAT	IOBTIO	IOBUSER	IOERBLOK
	IOERCCRA	IOERCCRL	IOERCSW	IOERDATA	IOERDW	IOEREXT	IOERLEN	IOERLOC	IOERSIZE	IOERSNSZ	IOERVSER	LOCK	MSSNEXT
	MSSSER	MSSTASK1	MSSVUA	NORET	OPERATOR	PROCIO	PSA	RCHADD	RCHELOK	RCUADD	RCUBLOK	RCUCHA	RCUPRIME
	RCUSUB	RCUTYPE	RDEVADD	RDEVBLOK	RDEVCUA	RDEVCUB	RDEVDED	RDEVDISA	RDEVFLAG	RDEVFTR	RDEVMOUT	RDEVOWN	RDEVSEL
	RDEVSER	RDEVSTAT	RDEVSYS	RDEVTYPE	READBUF	R0	R1	R10	R11	R12	R13	R14	R15
	R2	R3	R4	R5	R6	R7	R8	SAVEAREA	SAVEREGS	SILI	SYSVIRT	TYP2305	TYP2314
	TYP3330	TYP3340	UC	VIRTUAL	ZEROES								
DMKDSP	ACTIVTRQ	ADSPCH	AEXTSP	AFREE	AFRET	ALOKSP	ALOKVM	APSTAT1	APSTAT2	APSTAT3	APSTAT4	APTRAN	APUOPER
	ASYSVM	ATTN	BRING	CCC	CDC	CODE	CPAPRPND	CPCREG0	CPCREG6	CPEX	CPEXADD	CPEXBLOK	CPEXBPNT
	CPEXDEFR	CPEXFPNT	CPEXLPSW	CPEXPRIO	CPEXPROC	CPEXREGS	CPEXR10	CPEXR11	CPEXR15	CPEXR8	CPEXSIZE	CPEXTYPE	CPFRELK
	CPFRESW	CPMCHSE	CPMICON	CPPTLBR	CPRUN	CPSHRLK	CPSTATUS	CPSTAT2	CPSTAT3	CPSUPER	CPSYSLK	CPTIDLE	CPTIONT
	CPTPAGE	CPWAIT	CSW	CUE	C0	C1	C11	C13	C4	C5	C6	C7	C9
	DEFER	DISPATCH	DMKCCHRF	DMKCFMBK	DMKCVTBH	DMKERMSG	DMKFREE	DMKFREST	DMKFRET	DMKFRETL	DMKIOSER	DMKIOSRC	DMKLOKDF
	DMKLOKDS	DMKLOKPS	DMKLOKRL	DMKLOKSY	DMKLOKTR	DMKMCHSE	DMKMCTPR	DMKPERT	DMKPRGSM	DMKPTRAN	DMKPTRFD	DMKPTRFE	DMKPTRFP
	DMKPTRRC	DMKSCHDL	DMKSCHRL	DMKSCHTQ	DMKSCNVU	DMKSTKDE	DMKTRCEX	DMKTRCIO	DMKTRCIT	DMKTRCPG	DMKUSOFF	DMKVATAB	DMKVATAT
	DMKVATBC	DMKVATEX	DMKVATMD	DMKVIOMK	DMKVMASH	DMKVMCEX	ECBLOK	EMSINQSC	EMSMASK	EMSPEND	EMSPEXT	EMSPQUI	EMSREC
	EMSREXT	EMSRQUI	EXNPSW	EXOPSW	EXTCR0	EXTCR2	EXTCR4	EXTCR7	EXTMCDE	EXTPERAD	EXTPERCD	EXTSHCR0	EXTSHCR1
	FFS	FRLKPROC	F1	F255	F3	F4096	IDLEWAIT	IFCC	INTEX	INTEXF	INTPRL	INTTIO	IOBBPNT
	IOBCSW	IOBFLAG	IOBFPNT	IOBIRA	IOBLOK	IOBPAG	IOBUSER	IOMASK	IONPSW	IONTWAIT	IOOPSW	IPUADDRX	LASTUSER
	LOCK	LOCKSAV	LPUADDR	LPUADDRX	MFAMASK	MICBLOK	MICPEND	MICVIP	OFF	PAGEWAIT	PCI	PERADD	PERCODE
	PERMODE	PGADDR	PGBLOK	PGBSIZE	PGPNT	PGWAITIM	PREFIXA	PREFIXB	PRNPSW	PROBMODE	PROBSTRT	PROBTIME	PROCIO
	PROPSW	PSA	PWTPAGES	QUANTUM	QUANTUMR	RDEVBLOK	RDEVFICB	RUNCR0	RUNCR1	RUNPSW	RUNUSER	R0	R1
	R10	R11	R12	R13	R14	R15	R2	R3	R4	R5	R6	R7	R8
	R9	SAVE	SETUP	SIGDISP	SIGMASK	SIGWAKE	SIGXC	STACKVM	START	STOP	TEMPR0	TEMPSAVE	TIMEDISP
	TIMER	TRACCURR	TRACEND	TRACFLG2	TRACPROC	TRACSTRT	TRAC0A	TRAC0C	TRAC10	TRANMODE	TRCRUN	TRCUNBLK	TRCUNSTK

MODULE	EXTERNAL REFERENCES (LABELS AND MODULES)												
	TREXCR9	TREXFLAG	TREXIN1	TREXIN2	TREXNDSP	TREXT	TRQBFPNT	TRQBLOK	TRQBUSER	TRQBVAL	TYPE	UC	VCHADD
	VCHBLOK	VCHBUSY	VCHCEDEV	VCHCEPND	VCHCUINT	VCHCUTBL	VCHSEL	VCHSTAT	VCHTYPE	VCUADD	VCUBLOK	VCUCEPND	VCUCTCA
	VCUDVINT	VCUDVTBL	VCUINTS	VCUSHRD	VCUSTAT	VCUTYPE	VDEVADD	VDEVBLOK	VDEVCHAN	VDEVCSW	VDEVCUE	VDEVFLAG	VDEVINTS
	VDEVIOER	VDEVPEND	VDEVPOST	VDEVSTAT	VMAFF	VMAFFON	VMBLOK	VMCBLOK	VMCCSTAT	VMCCXINT	VMCF	VMCFPNT	VMCFREAD
	VMCFRUN	VMCFWAIT	VMCHSTRT	VMCHTBL	VMCOMP	VMCPNT	VMCPTIME	VMCPWAIT	VMCUSTRT	VMCXCODE	VMDEFSTK	VMDFTPNT	VMDSP
	VMDSTAT	VMDVSTRT	VMECEXT	VMESTAT	VMEXTCM	VMEXTPND	VMEXWAIT	VMFPRS	VMGPRS	VMIDLE	VMINQ	VMINVPAG	VMINVSEG
	VMIOACTV	VMIOINT	VMIOPND	VMIOWAIT	VMKILL	VMLOCK	VMLOGOFF	VMLSTPRC	VMMADDR	VMMCR6	VMMFE	VMMICRO	VMMNOSK
	VMMPROB	VMMSHADT	VMMVTMR	VMNDCNT	VMNEWCR0	VMNORUN	VMOSTAT	VMPAGES	VMPEND	VMPERCM	VMPERPND	VMPGPND	VMPGPNT
	VMPGWAIT	VMPRGIL	VMPRIDSP	VMPRRCT	VMPSTAT	VMPSW	VMPSWAIT	VMPXINT	VMQEPNT	VMQFPNT	VMQLEVEL	VMQSEND	VMQSTAT
	VMRON	VMRSTAT	VMRUN	VMSEG	VMSHADT	VMSHR	VMSTKCNT	VMSYSOP	VMTIMER	VMTLEVEL	VMTMINQ	VMTMOUTQ	VMTON
	VMTRBRIN	VMTRCTL	VMTREX	VMTREXT	VMTRIO	VMTRPER	VMTRPRG	VMTRPRV	VMTRSVC	VMTSEND	VMTTIME	VMVCR0	VMV370R
	VMWSPROJ	WAIT	WAITEND	WAITSTRT	XCDISP	XCMASK	XCPEND	XINTBLOK	XINTCODE	XINTMASK	XINTNEXT	XINTSIZE	XINTSORT
	XRIGHT16	XTNDLOCK	Y0	Y2	Y4	Y6	ZEROES						
DMKEIG	CCC	CCHCMDV	CCHDAV	CCHDI	CCHLOG80	CCHRCV	CCHREC	CCHUSV	COMPFES	COMPSEL	COMPSYS	CSW	FPS
	IFCC	IGBLAME	IGTERMSQ	IGVALIDB	INTERCCH	IOELPNTR	IOERBLOK	PSA	RTCODE0	RTCODE1	RTCODE2	RTCODE3	RTCODE4
	RTCODE5	R0	R1	R12	R13	R14	R15	R2	R3	R4	R9	SAVEAREA	SAVEWRK1
	SAVEWRK9	TERMSYS	TIOCCH										
DMKEMA	BUFFER	DEVICE	ERROR	FNAME	LINE								
DMKEMB	ERROR	SAVE	START	TIMER									
DMKEMC	ERROR	SYSTEM	VMCF										
DMKENT	ADSPCH	APSTAT1	AUTGO	DMKCVTAB	DMKDSPCH	DMKMIAEN	DMKMIAIN	DMKMIAKC	DMKMNIFI	DMKPRGMC	DMKSCHST	DMKSYSAT	F1
	F16	LOCK	MNCHSAMP	MNCHSIZE	MNCUBSY	MNDEVLEN	MNDEVLST	MNDVBSY	MN602ADD	MN602CHB	MN602CHQ	MN602CUB	MN602CUQ
	MN602DEV	MN602DLN	MN602DVQ	MN602HDR	MN602HLN	MN602SAM	MONCHPTR	MONCOM	MONDVLST	MONDVNUM	MONUTRB	PROCIO	PSA
	RCHADD	RCHBLOK	RCHQCNT	RCUADD	RCUBLOK	RCUBUSY	RCUCHA	RCUPRIME	RCUQCNT	RCUSTAT	RCUSUB	RCUTYPE	RDEVADD
	RDEVBLOK	RDEVBUSY	RDEVCUA	RDEVQCNT	RDEVSTAT	R0	R1	R11	R12	R13	R14	R15	R2
	R3	R4	R5	R6	R7	R8	R9	SAVEAREA	SAVEREGS	TRQBLOK	TRQBTOD	TRQBVAL	VMBLOK
DMKERM	AFREE	AFRET	ALARM	APTRAN	AQCNWT	ASYSVM	BLANKS	BRING	BUFCNT	BUFFER	BUFINLTH	BUFSIZE	COUNT
	C1	DEFER	DFRET	DMKCVTBD	DMKEMA00	DMKEMB00	DMKEMC00	DMKFREE	DMKFRET	DMKPTRAN	DMKQCNWT	DMKSYSRM	ERRMSG
	F2	LOCK	NORET	OPERATOR	PSA	R0	R1	R10	R11	R12	R13	R14	R15
	R2	R3	R4	R5	R6	R7	R8	R9	SAVEAREA	SAVEREGS	SAVER0	SAVER1	SAVER2
	SAVER3	SYSTEM	VMBLOK	VMSEG	XRIGHT16								
DMKEXT	ADSPCH	AFREE	APSTAT1	APSTAT3	APSTAT4	APUOPER	ASYSVM	CHAN0	CODE	CPAPRPND	CPCREG0	CPEXADD	CPEXBLOK
	CPEXREGS	CPEXR11	CPEXSIZE	CPLOKFL	CPRUN	CPSTATUS	CPSYSLK	CPWAIT	C0	C1	C2	C8	DMKCLKAP
	DMKCLKCC	DMKCLKSC	DMKCVTAB	DMKDSPCH	DMKDSPRU	DMKFREE	DMKLOKDF	DMKLOKSI	DMKLOKSY	DMKMCTMA	DMKMCTPR	DMKPSAER	DMKSTKMP
	ECBLOK	EMSPCLKC	EMSPEND	EMSPEXT	EMSPQUI	EMSPSHD	EMSPSYNC	EMSREC	EMSRSHD	EXOPSW	EXTCR8	EXTMASK	F256
	INTEX	IPUADDRX	LOCK	LPUADDR	MFAMASK	PREFIXA	PREFIXP	PSA	QUANTUM	QUANTUMR	RESET	RUNCR0	R0
	R1	R11	R12	R13	R14	R15	R2	R3	R4	R5	R6	SENSE	SIGEMS
	SIGSAVE	SIGSENSE	SIGXC	STOP	SYNCMASK	TIMEDISP	TIMER	TRACCURR	TRACEND	TRACFLG2	TRACPROC	TRACSTRT	TRAC13
	TRCSIGP	TYPE	VMBLOK	VMECEXT	VMGPRS	VMPSTAT	VMSEG	VMTMOUTQ	VMV370R	WAITEND	XCAPR	XCDISP	XCPEND

MODULE	EXTERNAL REFERENCES (LABELS AND MODULES)												
	XCRES	XCWAK											
DMKFCB	LINE	LOCK											
DMKFMT	ATTN	BUSY	CAW	CC	CD	CE	CONPARM	COUNT	CSW	CUE	DE	ERROR	EUA
	FLAG	IC	IONPSW	IOOPSW	NEWPSW	PROPSW	PSA	RA	REGSAV	RESET	R0	R1	R10
	R11	R12	R13	R14	R15	R2	R3	R4	R5	R6	R7	R8	R9
	SAVEAREA	SBA	SENSE	SF	SILI	SKIP	SM	START	TYPE	TYP3330	TYP3350	UC	UE
DMKFRE	ACORETBL	AEXTSP	AFREE	AFRET	ALOKSP	APSTAT1	APUOPER	ASYSVM	BALRSAVE	CODE	CORFPNT	CORPGPNT	CORTABLE
	COUNT	CPEXADD	CPEXBLOK	CPEXPROC	CPEXR0	CPEXSIZE	CPFRELK	CPFRESW	C2	DMKCPE	DMKDSPNP	DMKDSPRU	DMKLOKDF
	DMKLOKFR	DMKLOKSY	DMKPTRFR	DMKPTRFT	DMKQCNFT	DMKSTKLF	DMKSYSCS	DMKSYSRM	DMKVCNFT	FFS	FREENUM	FREER0	FREER1
	FREER14	FREER15	FREESAVE	FRLKPROC	F0	F1	F4096	IPUADDRX	LASTUSER	LOCK	LOCKSAV	LPUADDR	LPUADDRX
	PREFIXA	PROCIO	PSA	R0	R1	R10	R11	R12	R13	R14	R15	R2	R3
	R4	R5	R6	R7	R8	R9	SAVE	SAVESIZE	SIGWAKE	SIGXC	SIZE	TEMPSAVE	TRACCURR
	TRACEND	TRACFLG1	TRACPROC	TRACSTRT	TRAC67	TRCFREE	TRCFRET	TYPE	VMBLOK	VMLOCK	XPAGNUM	XTNDLOCK	
DMKGIO	ADSPCH	AFREE	AFRET	APTRAN	BRING	CLASDASD	CLASTAPE	CPEXADD	CPEXBLOK	CPEXFPNT	CPEXMISC	CPEXR0	CPEXSIZE
	CSW	C1	DEFER	DMKCCWTR	DMKDSPCH	DMKFREE	DMKFRET	DMKIOSQV	DMKPTRAN	DMKSCNVD	DMKSCNVU	DMKUNTFR	DMKUNTRN
	IL	IOBCAW	IOBCC3	IOBCLN	IOBCSW	IOBFATAL	IOBFLAG	IOBHVC	IOBIOER	IOBIRA	IOBLINK	IOBLOK	IOBMISC
	IOBMISC2	IOBSIZE	IOBSPEC2	IOBSTAT	IOERBLOK	IOERCSW	IOERDATA	IOEREXT	IOERSIZE	LOCK	PSA	R0	R1
	R10	R11	R12	R13	R14	R15	R2	R3	R4	R5	R6	R7	R8
	R9	SAVEAREA	SAVEREGS	SAVER2	TYP3340	UE	VDEVBLOK	VDEVBUSY	VDEVCHAN	VDEVCPEX	VDEVCSW	VDEVDED	VDEVFLAG
	VDEVFLG2	VDEVIOB	VDEVIOER	VDEVPEND	VDEVPOSN	VDEVRELN	VDEVRES	VDEVRRE	VDEVRRF	VDEVSTAT	VDEVTYPC	VDEVUC	VMACTDEV
	VMBLOK	VMCOMP	VMDVSTRT	VMESTAT	VMEXTCM	VMEXWAIT	VMGPRS	VMIDLE	VMIOCNT	VMIOWAIT	VMLOPRI	VMPSW	VMQLEVEL
	VMRSTAT	VMSEG	VRRBLOK	VRRCPEX	VRRRES	VRRSTAT							
DMKGRF	ADSPCH	AFREE	AFRET	ALARM	APSTAT1	APUOPER	ARIOCH	ARIODV	ASYSVM	ATTN	ATTR2	ATTR457	ATTR7
	BLANKS	BRING	BUFAPL	BUFCNT	BUFFER	BUFINLTH	BUFSIZE	CC	CCC	CD	CDC	CE	CHC
	CLASGRAF	CONACTV	CONADDR	CONCCW1	CONCCW2	CONCCW4	CONCNT	CONDATA	CONDWC	CONESCP	CONOUTPT	CONPARM	CONPNT
	CONRESP	CONRETN	CONSTAT	CONSYNC	CONTASK	CONTSIZE	CONTSKSZ	CPEXADD	CPEXBLOK	CPEXR0	CPEXSIZE	CPID	C1
	DE	DEFER	DMKBLDVM	DMKCFMAT	DMKCFMBK	DMKCFMEN	DMKCNSED	DMKCPIEM	DMKCVTAB	DMKCVTBD	DMKCVTHB	DMKDSPCH	DMKFREE
	DMKFRET	DMKGRTAB	DMKGRTAI	DMKGRTBL	DMKGRTDS	DMKGRTFD	DMKGRTFM	DMKGRTFO	DMKGRTIN	DMKGRTPF	DMKGRTP6	DMKIOERR	DMKIOEST
	DMKIOSQR	DMKLOKSW	DMKMSWR	DMKPTRAN	DMKQCNCL	DMKQCNET	DMKQCNTO	DMKQCNWT	DMKRIOCN	DMKSCHRT	DMKSCHST	DMKSCNRD	DMKSCNRU
	DMKSTRCP	DMKSTKMP	DMKSYSNM	DMKTBLGL	DMKTBLUP	DMKTBMTI	DMKTBMTO	DMKTBMZI	DMKTBMZO	EDIT	ETX	EUA	F0
	F1	F255	F256	F3	F4	F5	F8	GRTBLOK	GRTCLRCP	GRTCLRDS	GRTCLRIA	GRTCLRL	GRTCPPCP
	GRTCPPDS	GRTCPPL	GRTCPRCP	GRTCPRDS	GRTCPRL	GRTCRDCP	GRTCRDDS	GRTCRDL	GRTEWRCP	GRTHLDCP	GRTHLDDS	GRTHLDL	GRTINHDS
	GRTINHL	GRTMORCP	GRTMORDS	GRTMORL	GRTMRDCP	GRTMRDDS	GRTMRDL	GRTNACCP	GRTNACDS	GRTNACL	GRTRMICP	GRTRMIDS	GRTRMIL
	GRTRUNCP	GRTRUNDS	GRTRUNL	GRTRUNRL	GRTVMPCP	GRTVMPDS	GRTVMPL	GRTVMRCP	GRTVMRDS	GRTVMRL	GRTWINCP	GRTWINDS	GRTWINL
	GRTWRTCP	GRTWRTDS	GRTWRTL	IC	IFCC	INHIBIT	INTREQ	IOBCAW	IOBCCPY	IOBCSW	IOBERP	IOBFATAL	IOBFLAG
	IOBIOER	IOBIRA	IOBLINK	IOBLOK	IOBMISC	IOBMISC2	IOBRADD	IOBRCNT	IOBSENS	IOBSIZE	IOBSPEC	IOBSTAT	IOBUNSL
	IOBUSER	IOERBLOK	IOERCSW	IOERDATA	IOEREXT	IOERFLG3	IOERNUM	IOERREAD	IOERSIZE	LOCK	LOGDROP	LOGHOLD	MNCLRESP
	MNCOERD	MSGNUM	NORET	NOTEXT	NOTIME	NOTRESP	PREFIXA	PRGC	PRIORITY	PROCIO	PRTC	PSA	RA
	RCUBLOK	RCUDVTBL	RCUPRIME	RCUSUB	RCUTYPE	RDEVACTV	RDEVAIOB	RDEVAIRA	RDEVAPLP	RDEVBLOK	RDEVCON	RDEVCORD	RDEVCPNA
	RDEVCTL	RDEVCUA	RDEVDED	RDEVDISA	RDEVDISB	RDEVDROP	RDEVENAB	RDEVFLAG	RDEVGRTE	RDEVGRTY	RDEVHIO	RDEVHOLD	RDEVIOER
	RDEVLOG	RDEVMORE	RDEVREAD	RDEVRUN	RDEVSTAT	RDEVSTA2	RDEVTEXT	RDEVTFLG	RDEVTMCD	RDEVTRQ	RDEVTYPC	RDEVTYPE	RDEVUSER

MODULE	EXTERNAL REFERENCES (LABELS AND MODULES)												
	READBUF	R0	R1	R10	R11	R12	R13	R14	R15	R2	R3	R4	R5
	R6	R7	R8	R9	SAVEAREA	SAVER0	SAVER2	SBA	SF	SILI	SVMSTAY	SYSTEM	TEMPSAVE
	TIMEDISP	TRQBIRA	TRQBLOK	TRQBSIZE	TRQBUSER	TRQBVAL	TYP3066	TYP3277	TYP3278	TYP3284	UC	UCASE	UE
	VCONCTL	VCONRBSZ	VCONRBUF	VCONRCNT	VDEVBLOK	VDEVCON	VMBLOK	VMCF	VMCFWAIT	VMDVSTRT	VMGENIO	VMGRFTAB	VMLOGOFF
	VMLOGON	VMMCPENV	VMMLEVEL	VMMLINED	VMOSTAT	VMPA2APL	VMPFUNC	VMPXINT	VMQSTAT	VMRSTAT	VMSEG	VMSYSOP	VMTLEND
	VMVTERM	WCC0	WCC3	WCC4	WCC5	WCC6	XINTBLOK	XINTCODE	XINTNEXT	XINTSIZE	XINTSORT	XTNDLOCK	ZEROES
DMKGRT	AFREE	APTRAN	ASYSVM	ATTR2	ATTR457	ATTR7	BALRSAVE	BALR2	BRING	BUFAPL	BUFFER	CC	CD
	C1	DEFER	DMKBOXBX	DMKFREE	DMKGRFMT	DMKPTRAN	ETX	EUA	F4	GRTCLRDS	GRTCLRL	GRTCPPDS	GRTCPPL
	GRTCPRDS	GRTCPRL	GRTCRDDS	GRTCRDL	GRTHLDDS	GRTHLDL	GRTINHDS	GRTINHL	GRTMORDS	GRTMORL	GRTMRDDS	GRTMRDL	GRTNACDS
	GRTNACL	GRTRMIDS	GRTRMIL	GRTRUNDS	GRTRUNL	GRTRUNRL	GRTVMPDS	GRTVMPL	GRTVMRDS	GRTVMRL	GRTWINDS	GRTWINL	GRTWRTDS
	GRTWRTL	IC	PSA	RA	RDEVBLOK	RDEVGRTY	R0	R1	R11	R12	R13	R14	R15
	R2	R3	R4	R5	R6	R7	R8	SAVEAREA	SAVEREGS	SBA	SF	SILI	SYSTEM
	VMBLOK	VMGRFTAB	VMSEG	WCC0	WCC3	WCC4	WCC56	WCC6					
DMKGRW	ATTR2	ATTR457	ATTR7	BUFAPL	CC	CD	ETX	EUA	GRTCLRDS	GRTCLRL	GRTCPPDS	GRTCPPL	GRTCPRDS
	GRTCPRL	GRTCRDDS	GRTCRDL	GRTHLDDS	GRTHLDL	GRTINHDS	GRTINHL	GRTMORDS	GRTMORL	GRTMRDDS	GRTMRDL	GRTNACDS	GRTNACL
	GRTRMIDS	GRTRMIL	GRTRUNDS	GRTRUNL	GRTRUNRL	GRTVMPDS	GRTVMPL	GRTVMRDS	GRTVMRL	GRTWINDS	GRTWINL	GRTWRTDS	GRTWRTL
	IC	RA	SBA	SF	SILI	WCC0	WCC3	WCC4	WCC6				
DMKHVC	ADSPCH	AFREE	AFRET	APTRAN	ASYSVM	BLANKS	BRING	CCC	CDC	CE	CHC	CPUID	C1
	DE	DEFER	DMKCCWTC	DMKCCWTR	DMKCFGCL	DMKCFMBK	DMKCFMEN	DMKCVTDT	DMKDGDDK	DMKDSPCH	DMKFREE	DMKFRET	DMKGIOEX
	DMKHVDAL	DMKPGSSS	DMKPRGSM	DMKPSASP	DMKPTRAN	DMKSCNVU	DMKSSSHV	DMKTMRPT	DMKUNTFR	DMKVMCFC	DMKVSIEX	FFS	F1
	F16	F256	F4	F4095	F4096	F60	IFCC	IL	IOBCAW	IOBCSW	IOBLOK	IOBRADD	IOBSIZE
	LOCK	PCI	PRGC	PRTC	PSA	RCWADDR	RCWCCW	RCWCTL	RCWHEAD	RCWPNT	RCWRCNT	RCWTASK	R0
	R1	R10	R11	R12	R13	R14	R15	R2	R3	R4	R5	R6	R7
	R8	R9	TEMPR6	TEMPR8	UC	UE	VDEVBLOK	VDEVIOB	VMBLCK	VMCF	VMCFWAIT	VMCOMND	VMCONBUF
	VMCONLN	VMCPLEN	VMDVSTRT	VMESTAT	VMEXTCM	VMEXWAIT	VMFSTAT	VMGPRS	VMINST	VMIOWAIT	VMMCODE	VMMLEVEL	VMMTEXT
	VMNPWOCL	VMOSTAT	VMPRIDSP	VMPSW	VMQSTAT	VMRSTAT	VMSEG	VMSLEEP	VMSTOR	VMTIME	VMVIRCF	XPAGNUM	ZEROES
DMKHVD	ACCTACNO	ACCTBLOK	ACCTDIST	ACCTLENG	ACCTUSER	ACNTBLOK	ACNTCODE	ACNTDATA	ACNTNUM	ACNTSIZE	ACNTUSER	AFREE	AFRET
	APSTAT1	APTRAN	ASYSVM	BRING	CLASGRAF	CLASSPEC	CLASTERM	CLASURI	CLASURO	CPUMCELL	CPUVERSN	C1	DEFER
	DMKACOQU	DMKCPEID	DMKCPEPP	DMKCPVAA	DMKCVTDB	DMKDRDER	DMKDRDMP	DMKDRDSY	DMKFREE	DMKFRET	DMKIOEFM	DMKIOEFR	DMKIOEHS
	DMKPSASP	DMKPTRAN	DMKQNTBL	DMKRPAGT	DMKRPAPT	DMKSCNRU	DMKSCNVD	DMKSCNVS	DMKSCNVU	DMKSNCP	DMKSYSCT	DMKSYSER	DMKSYSRM
	DMKUDRDS	DMKUDRFU	DMKUDRMD	DMKUDRRV	DMKUDUMN	FFS	FTR35MB	F1	F16	F2	F20	F24	F256
	F4	F4095	F4096	F60	F8	IPUADDR	LOCK	NICBLOK	NICGRAF	NICLLEN	NICSIZE	NICTMCD	NICTYPE
	NPRCNT	NPRNAME	NPRPAGCT	NPRPNT	NPRSTART	NPRTBL	NPRVOL	PREFIXA	PROCIO	PSA	RDEVBLOK	RDEVCODE	
	RDEVFLAG	RDEVFTR	RDEVLLEN	RDEVMDL	RDEVNICL	RDEVOWN	RDEVTMCD	RDEVTYPC	RDEVTYPE	R0	R1	R10	R11
	R12	R13	R14	R15	R2	R3	R4	R5	R6	R7	R8	R9	SAVEAREA
	SAVEREGS	SAVER0	SYSIPLDV	SYSTEM	TYPBSC	TYP2305	TYP2314	TYP2319	TYP3210	TYP3277	TYP3278	TYP3330	TYP3340
	TYP3350	UDBFBLOK	UDBFSIZE	UDBFVADD	UDIRBLOK	UDIRDISP	UDIRUSER	UMACACCT	UMACELCK	VDEVBLOK	VDEVDED	VDEVREAL	VDEVSTAT
	VDEVTYPC	VDEVTYPE	VMACCOUN	VMACOUNT	VMBLOK	VMCLASSA	VMCLASSB	VMCLASSC	VMCLASSE	VMCLASSF	VMCLEVEL	VMDVSTRT	VMESTAT
	VMEXTCM	VMGPRS	VMINST	VMPA2APL	VMPSTAT	VMPSW	VMPSWDCT	VMQSTAT	VMSEG	VMSIZE	VMTERM	VMTRMID	VMUSER
	VMVTERM	VMV370R	XPAGNUM	XRIGHT24	X2048BND	ZEROES							
DMKIMG	BUFSIZE	CURRSAVE	DECAREA	DECDCBAD	EGPR0	EGPR15	EGPR8	LOCCNT	MAINHIGH	NUCON	R0	R1	R12

MODULE	EXTERNAL REFERENCES (LABELS AND MODULES)												
	R13	R14	R15	R2	R3	R4	R5	R6	R7	R8	R9	SAVEAR	SSAVE
	STRTADDR	TEXT											
DMKIOC	CLASDASD	CLASTERM	DEVCODE	OBRDEVSH	OBRDEVTN	OBRRECN	OBRSHCBR	OBRSWSN	PSA	RCUBLOK	RCUTYPE	RCU2701	RCU2702
	RDEVBLOK	RDEVCUA	RDEVMDL	RDEVNAME	RDEVSADN	RDEVTMCD	RDEVTYPC	RDEVTYPE	R0	R11	R12	R13	R4
	R5	R6	R8	R9	SAVEAREA	SAVEREGS	TYP2305	TYP2311	TYP3330	ZEROES			
DMKIOE	ADSPCH	AFREE	AFRET	APSTAT1	ASYSVM	CCC	CDC	CLASDASD	CLASTAPE	CLASURO	CONCCW3	CONDATA	CONDCNT
	CPEXADD	CPEXBLOK	CPEXFPNT	CPEXREGS	CPEXSIZE	DMKCCHRT	DMKCVTAB	DMKDSPCH	DMKFREE	DMKFRET	DMKIOFC1	DMKIOFIN	DMKIOFM1
	DMKIOFOB	DMKIOFST	DMKIOFVR	DMKIOGF1	DMKIOGF2	DMKSTKCP	DMKSYSTZ	ERRBLOK	ERRCCNT	ERRCCW	ERRCORR	ERRHEADR	ERRIOB
	ERRIOER	ERRKEY	ERRMIOB	ERRMIOER	FRRMSIZE	ERRPARM	ERRSDR	ERRSIZE	ERRVCLID	FTREXTSN	F1	F255	F4
	F7	F8	IFCC	IOBCP	IOBFATAL	IOBFLAG	IOBHVC	IOBIOER	IOBLCK	IOBRADD	IOBSTAT	IOERBLOK	IOERCCRA
	IOERCCRL	IOERCEMD	IOERCSW	IOERDATA	IOEREXT	IOERFLG2	IOERPNT	ICERSIZE	IRMAND	IRMBIT1	IRMBIT2	IRMBLOK	IRMBYT1
	IRMBYT2	IRMFLG	IRMLMT	IRMLMTCT	IRMMAXCT	IRMOR	IRMRLADD	IRMSIZE	LOCK	PROCIO	PSA	RDEVBLOK	RDEVCTRS
	RDEVFTR	RDEVIOER	RDEVIRH	RDEVSER	RDEVSTAT	RDEVTYPC	RDEVTYPE	R0	R1	R10	R11	R12	R13
	R14	R15	R2	R3	R4	R5	R6	R7	R8	R9	SAVEAREA	SAVEREGS	SAVER1
	SAVER11	SDRBLOK	SDRBSIZE	SDRCTRS	SDRFLAGS	SDRLNGTH	SDRSHRT	TIMEDISP	TYP2305	TYP3330	TYP3340	TYP3350	TYP3410
	TYP3420	UC	VMBLOK	VMCLASSF	VMCLEVEL	ZEROES							
DMKIOF	AFREE	AFRET	APTRAN	BRING	CDC	CLASDASD	CLASGRAF	CLASSPEC	CLASTAPE	CLASTERM	CLASURI	CLASURO	CPEXBLOK
	CPEXFPNT	CPEXREGS	CPEXR6	CPEXSIZE	CPUID	CPUVERSN	DEFER	DMKERMSG	DMKFREE	DMKFRET	DMKIOCVT	DMKIOECQ	DMKIOECT
	DMKIOEEP	DMKIOEES	DMKIOEIQ	DMKIOEMQ	DMKIOEMX	DMKIOENI	DMKIOENQ	DMKIOERP	DMKIOERQ	DMKIOESQ	DMKIOEVQ	DMKPGTVG	DMKPGTVR
	DMKPTRAN	DMKPTRUL	DMKRPAGT	DMKRPAPT	DMKSTKCP	DMKSYSER	ERRBLOK	ERRCCNT	ERRCCW	ERRCONT	ERRCORR	ERRIOB	ERRIOER
	ERRKEY	ERRMIOB	ERRMIOER	ERRPARM	ERRSDR	ERRVOLID	FFS	FTREXTSN	F15	F255	F4	F7	F8
	IOBFATAL	IOERADR	IOERBLOK	IOERCSW	IOERDATA	IOEREXT	IOERFLG3	ICERLEN	IOERPNT	IOERREAD	IOERSIZE	IOERVSER	LASTCYL
	LOCK	MDRREC	OBRCORL	OBRCPIDN	OBRCSWN	OBRCUAIN	OBRCUAPR	OBRDDCNT	OBRECD	OBRFCCWN	OBRHAN	OBRIORTY	OBRKEYN
	OBRLSKN	OBRPGMN	OBRRECN	OBRSDRCT	OBRSDRSH	OBRSHOBR	OBRSNSCT	OBRSSDR1	OBRSWSN	OBRTEMP	OBRVOLN	OBR33SNS	PSA
	RDEVBLOK	RDEVCTRS	RDEVFTR	RDEVMDL	RDEVTYPC	RDEVTYPE	RECCCPD	RECFLAG1	RECNXT	RECPAG	RECPAGFL	RECPAGIU	R0
	R1	R10	R11	R12	R13	R14	R15	R2	R3	R4	R5	R6	R7
	R8	R9	SAVEAREA	SAVEREGS	SDRBLOK	SDRCTRS	SDRCUA	SDRFLAGS	SDRFLCT	SDRLNGTH	SDRMAX	SDROVFWK	SDRPRMCT
	SDRRDEV	SDRRECD	SDRSHRT	SDRSIZE	SDRSIZE1	SYSTEM	TNSCPIDN	TNSDEVAD	TNSKEYN	TNSREC	TNSSNS1	TNSSWS3	TNSVOLID
	TYPBSC	TYPTTY	TYP1050	TYP1403	TYP1443	TYP2305	TYP2501	TYP2520R	TYP2540R	TYP2700	TYP2741	TYP3066	TYP3203
	TYP3210	TYP3211	TYP3330	TYP3340	TYP3350	TYP3410	TYP3420	TYP3505	TYP3800	VMBLOK	VMUSER	XOBRFLAG	XOBRT1
	XOBRT3	XOBR010	XOBR150	XOBR180	XOBR512	ZEROES							
DMKIOG	ADSPCH	AFREE	AFRET	AMCHAREA	APSTAT1	APSTAT2	APTRLK	ARIOCH	ARIOCT	ASYSVM	BRING	CE	CHANID
	CPEXSIZE	CPUID	CPUMCELL	CPUMODEL	CPUVERSN	C1	DE	DEFER	DMKCCHCF	DMKCCHMX	DMKCCHSZ	DMKCCH60	DMKCVTAB
	DMKDSPCH	DMKEIG80	DMKERMSG	DMKFREE	DMKFRET	DMKIOECT	DMKIOEEP	DMKIOEES	DMKIOEFR	DMKIOEHS	DMKIOEMX	DMKIOENI	DMKIOSQR
	DMKPGTVG	DMKPGTVR	DMKPTRAN	DMKPTRLK	DMKPTRUL	DMKRIODV	DMKRICSF	DMKRPAGT	DMKRPAPT	DMKSCNRU	DMKSEV70	DMKSIX60	DMKSYSCT
	DMKSYSER	DMKSYSTZ	ECSWLOG	FFS	F0	F1	F255	F7	IOBCAW	IOBCC2	IOBCSW	IOBFATAL	IOBIRA
	IOBLINK	IOBLOK	IOBRADD	IOBSIZE	IOBSTAT	IOBUSER	IOELPNTR	IPUADDR	LASTCYL	LOCK	MCHAREA	MCHCPEX	MCHDAMLN
	MCHFIX	MCHLEN1	MCHMODEL	MCHPROCA	MCHREC	MCNPSW	MODEL135	MODEL138	MODEL145	MODEL148	MODEL155	MODEL158	MODEL165
	MODEL168	MOD3031	MOD3032	MOD3033	MOD4331	MOD4341	NOMODEL	PROCIO	PSA	RCHBLOK	RCHSTIDC	RCHTYPE	RCH370
	RDEVBLOK	RDEVCODE	RDEVTYPE	RECCCPD	RECFLAG1	RECFLAG2	RECMODE	RECNXT	RECPAG	RECPAGDN	RECPAGFA	RECPAGFL	RECPAGFM
	RECPAGFR	RECPAGIU	R0	R1	R10	R11	R12	R13	R14	R15	R2	R3	R4
	R5	R6	R7	R8	R9	SAVEAREA	SAVEREGS	SAVEWRK2	SAVEWRK7	SYSIPLDV	SYSTEM	TYP2305	TYP3330

MODULE	EXTERNAL REFERENCES (LABELS AND MODULES)												
	TYP3340	TYP3350	UE	VMBLOK	VMSEG	WAIT							
DMKIOS	ADSPCH	AEXTSP	AFREE	AFRET	ALOKSP	APSTAT1	APUCPER	ASYSVM	ATTN	BUSY	CAW	CC	CCC
	CDC	CE	CHC	CLASDASD	CLASGRAF	CLASSPEC	CLASTAPE	CLASTERM	CLASURI	CLASURO	CMDREJ	CODE	COUNT
	CPCREG0	CPCREG8	CPEXADD	CPEXBLOK	CPEXPROC	CPEXR0	CPEXR1	CPEXR13	CPEXSIZE	CPINITD	CPRUN	CPSTATUS	CPSUPER
	CPWAIT	CSW	CUE	C0	C1	C2	C8	DE	DMKBSCER	DMKCCHIS	DMKCCHNT	DMKCNSIN	DMKDASER
	DMKDIAIR	DMKDSBRD	DMKDSPA	DMKDSPCH	DMKDSPRU	DMKENTSK	DMKFREE	DMKFRET	DMKGRFIN	DMKIOERR	DMKLOKDF	DMKLOKSY	DMKRGAIN
	DMKRIOCN	DMKRIOCT	DMKRIOCU	DMKRNHIN	DMKRSPER	DMKRSPEX	DMKSCHDL	DMKSCNRU	DMKSSSI1	DMKSSSI2	DMKSTKIO	DMKSTKLF	DMKSTKMP
	DMKTAPER	DMKTRDSI	DMKVIOIN	FFS	FIRRPS	F0	F1	F16	F2	F8	IFCC	IL	INTREQ
	INTTIO	IOBBPNT	IOBCAW	IOBCC1	IOBCC2	IOBCC3	IOBCP	IOBCSW	IOBCYL	IOBERP	IOBFATAL	IOBFLAG	IOBFLT
	IOBFPNT	IOBHIO	IOBHVC	IOBIMSTK	IOBIOER	IOBIRA	IOBLINK	IOBLOK	IOBMINI	IOBPAG	IOBPATHF	IOBRADD	IOBRCAW
	IOBRELCU	IOBRES	IOBRETRY	IOBRSTRT	IOBSIOF	IOBSIZE	IOBSNSIC	IOBSPEC	IOBSPEC2	IOBSPLT	IOBSTAT	IOBTIO	IOBUC
	IOBUNSL	IOBUSER	IOBVADD	IOERBLOK	IOERCCW	IOERCSW	IOERDATA	IOEREXT	IOERLEN	IOERSIZE	IOERSNSZ	IOOPSW	IPUADDRX
	LASTUSER	LOCK	LOCKSAV	LPUADDR	MNCLSEEK	MNCOCYL	PCI	PREFIXA	PREFIXE	PRGC	PROCIO	PRTC	PSA
	QUANTUMR	RCHADD	RCHBLOK	RCHBMX	RCHBUSY	RCHFIOB	RCHMPX	RCHQCNT	RCHRSTQ	RCHSEL	RCHSTAT	RCHTYPE	RCH370
	RCUADD	RCUBLOK	RCUBUSY	RCUCHA	RCUCHAOF	RCUCHBOF	RCUCHCCF	RCUCHD	RCUCHDOF	RCUDISA	RCUFIOB	RCUPRIME	RCUQCNT
	RCURSTQ	RCUSCED	RCUSHRD	RCUSTAT	RCUSUB	RCUTYPE	RCWCCNT	RCWTASK	RDEVALD	RDEVAIOB	RDEVALT	RDEVATT	RDEVBLOK
	RDEVBUCH	RDEVBUSY	RDEVCONC	RDEVCUA	RDEVCUB	RDEVCYL	RDEVDED	RDEVDISA	RDEVFIOB	RDEVFLAG	RDEVFTR	RDEVIOBL	RDEVIOCT
	RDEVIOER	RDEVLIOB	RDEVNRDY	RDEVQCNT	RDEVRACT	RDEVSCED	RDEVSKUP	RDEVSTAT	RDEVSTA2	RDEVTYPC	RDEVTYPE	RDEVUSER	RDEV333V
	RUNUSER	R0	R1	R10	R11	R12	R13	R14	R15	R2	R3	R4	R5
	R6	R7	R8	R9	SAVE	SAVEAREA	SAVEREGS	SAVER11	SAVEWRK2	SEEKCL	SIGWAKE	SIGXC	SILI
	SKIP	SM	START	SYSVIRT	TEMPR9	TEMPSAVE	TIMEDISP	TIMER	TRACEEF	TRACCURR	TRACEND	TRACFLG1	TRACFLG2
	TRACPROC	TRACSTRT	TRAC05	TYPBSC	TYPCTCA	TYPE	TYP2305	TYP3330	TYP3800	UC	UE	VDEVBLOK	VDEVIOCT
	VDEVREAL	VIRTUAL	VMBLOK	VMDSTAT	VMESTAT	VMEXTCM	VMFPRS	VMGPRS	VMIDLE	VMINQ	VMIOWAIT	VMLOCK	VMLOCKER
	VMPSW	VMRSTAT	VMRUN	VMSEG	VMTMOUTQ	VMTRCTL	VMTRSIO	WAITEND	XTNDLOCK	Y0	Y2	Y4	Y6
DMKISM	AFREE	CD	C1	DMKFREE	DMKPTRAN	DMKPTRUL	DMKUNTIS	F16	F2	F4	F8	IDA	IOBCAW
	IOBIRA	IOBLOK	IOBMISC	PSA	RCWCCNT	RCWCCW	RCWIO	RCWPNT	RCWRCNT	RCWTASK	RCWVCAW	R0	R1
	R10	R11	R12	R13	R14	R15	R2	R3	R4	R5	R6	R7	R8
	R9	SAVEAREA	SAVEREGS	VMBLOK	VMSEG	XPAGNUM							
DMKJRL	ACNTBLOK	ACNTSIZE	AFREE	AFRET	ALARM	APSTAT1	APUOPER	AQCNWT	ASYSOP	BLANKS	CL	DFRET	DMKACOQU
	DMKCVTBD	DMKCVTBH	DMKCVTDT	DMKERMSG	DMKFREE	DMKFRET	DMKLOKSW	DMKQCNWT	DMKSCNAU	DMKSCNFD	DMKSCNRD	DMKSYSJR	F0
	JPSCBLOK	JPSLNKAR	JPSLNKMS	JPSLNKU	JPSLOGAR	JPSLOGMS	JPSLOGU	JPSPCHN	JRLSQCK	LINKJRL	LOCK	LOGONJRL	NORET
	PSA	PWDCHAIN	PWDDATE	PWDIBLOK	PWDINVCT	PWDLOG	PWDSIZE	PWDTERMA	PWDUSRID	R0	R1	R10	R11
	R12	R13	R14	R15	R2	R3	R4	R5	R6	R7	R8	R9	SAVEAREA
	SAVEREGS	SAVER0	TIMEDISP	VMACNT	VMBLOK	VMDISC	VMKILL	VMOSTAT	VMPSWICT	VMTERM	VMUSER		
DMKLDOO	AFREE	AFRET	APOINT	APSV	BLANKS	BRAD	CAW	CCW1	CMD	CSW	CTL	C2	DE
	DEVICE	DMKCPE	DMKPSA	DMKWRM	ERRCCW	ESIDTB	EXNPSW	FSIZE	IOMASK	IPLPSW	LOCCT	NOP	NUM
	OUTPUT	PLIST	PSW	RDCCW	READ	RETREG	RETT	R0	R1	R10	R12	R13	R14
	R15	R2	R3	R4	R5	R6	R7	R8	R9	SILI	SIZE	SPEC	START
	TBLCT	TBLREF	TEMPST	TMPLOC	TYPTR	UC	WRITE						
DMKLNK	AFREE	AFRET	AQCNWT	BLANKS	BUFFER	BUFINLTH	BUFNXT	BUFSIZE	CLASDASD	CPEXBLOK	CPEXR1	DMKCFPRD	DMKCVTBD
	DMKCVTBH	DMKCVTHB	DMKEPSWD	DMKERMSG	DMKFREE	DMKFRET	DMKJRLIL	DMKJRLSL	DMKLOCK	DMKLOCKD	DMKQCNRD	DMKQCNWT	DMKSCNAU

MODULE	EXTERNAL REFERENCES (LABELS AND MODULES)												
	DMKSCNFD	DMKSCNLI	DMKSCNVN	DMKSCNVS	DMKSCNVU	DMKSSSLN	DMKSSSMQ	DMKSTKCP	DMKSYSJR	DMKUDRFD	DMKUDRFU	DMKUDRRV	DMKVDSLK
	EDIT	ERRMSG	FFS	FTR2311B	FTR2311T	F1	F15	F2	F4	F4095	F7	F8	INHIBIT
	JPSCBLOK	JPSLNKDS	LINKJRL	LOCK	MASKLINK	MSSFLAGS	MSSNEXT	MSSPRES	MSSSIZE	MSSTASK1	MSSTASK3	NORET	NOTRESP
	PSA	PSAMSS	RDEVBLOK	RDEVFTR	RDEVTYPC	RDEVTYPE	READ	R0	R1	R10	R11	R12	R13
	R14	R15	R2	R3	R4	R5	R6	R7	R8	R9	SAVEAREA	SAVEREGS	SAVERETN
	SAVER0	SAVER1	SAVER11	SAVER2	SAVESIZE	SAVEWRK1	SAVEWRK2	SAVEWRK4	SAVEWRK5	SAVEWRK6	SAVEWRK7	SAVEWRK8	SAVEWRK9
	SYSVIRT	TYP2311	TYP2314	TYP3330	UCASE	UDBFBLOK	UDBFSIZE	UDBFVADD	UDEVADD	UDEVBLOK	UDEVDED	UDEVDISP	UDEVFTR
	UDEVLINK	UDEVLKDV	UDEVLKID	UDEVLM	UDEVLONG	UDEVLR	UDEVLW	UDEVMODE	UDEVNCYL	UDEVPASR	UDEVR	UDEVRELN	UDEVSTAT
	UDEVTDSK	UDEVTYPC	UDEVTYPE	UDEVVSER	UDEVW	UDIRBLOK	UDIRDISP	VCHBLOK	VCHDED	VCHSTAT	VDEVBLOK	VDEVBND	VDEVFLAG
	VDEVRDO	VDEVREAL	VDEVRELN	VDEVTYPC	VDEVTYPE	VDEVUSER	VIRTUAL	VMBLOK	VMCF	VMCOMND	VMFSTAT	VMKILL	VMLOGON
	VMNPWOCL	VMOSTAT	VMPSTAT	VMPSWDCT	VMRSTAT	VMUSER	VMVIRCF	VMV370R	WRITE	ZEROES			
DMKLOC	ADSPCH	AFREE	AFRET	ASYSLC	BALRSAVE	BALR14	CPEXADD	CPEXBLOK	CPEXFENT	CPEXPROC	CPEXREGS	CPEXSIZE	DMKDSPCH
	DMKFREE	DMKFRET	DMKSTKMP	DMKSTKOP	DMKSYSLB	LOCKBLOK	LOCKNAME	LOCKNEXT	LOCKQUE	LOCKSIZE	LPUADDR	PSA	R0
	R1	R10	R12	R14	R15	R2	R3	R4	R5	R6	R7	R8	R9
	SYSLOCS												
DMKLOG	ADSPCH	AFREE	AFRET	APSTAT1	APUOPER	ARIODC	ARIODV	ASYSLC	ASYSOP	ASYSVM	AVMREAL	BLANKS	CLASDASD
	CLASSPEC	CLASTERM	CPASTAVL	CPASTON	CPEXSIZE	CPMICAVL	CPSTAT2	CPUID	DMKACCN	DMKBLDEC	DMKBLDRT	DMKCFGII	DMKCVTBH
	DMKDSPCH	DMKEPSWD	DMKERMSG	DMKFREE	DMKFRERC	DMKFRET	DMKJRLLO	DMKLNKSB	DMKLOHON	DMKLOKSW	DMKQCNSY	DMKSCHRT	DMKSCH80
	DMKSCNAU	DMKSCNFD	DMKSCNRD	DMKSCNRU	DMKSCNVD	DMKSCNVN	DMKSCNVS	DMKSCNVU	DMKSSSL1	DMKSSSL2	DMKSSSL3	DMKSSSMQ	DMKSTKCP
	DMKSYSJR	DMKSYSMA	DMKSYSNM	DMKUDRFU	DMKUDRMD	DMKUDRRD	DMKUDRRV	DMKVDSAT	DMKVDSDF	FFS	F4095	F7	F8
	INHIBIT	IOBLOK	IOBUSER	JPSCBLOK	JPSLOGDS	LOCK	LOGONJRL	LPUADDR	LPUADDRX	MASKLOG	MICBLOK	MICCREG	MICEVMA
	MICRSEG	MICSIZE	MICVPSW	MICVTMR	MICWORK	MSSFLAGS	MSSNEXT	MSSSIZE	MSSTASK3	MSSVUA	NEWPAGES	NEWSEGS	NICBLOK
	NICFLAG	NICPSUP	NICSIZE	NICUSER	PREFIXB	PROCIO	PSA	RDEVADD	RDEVAIOB	RDEVBLOK	RDEVDED	RDEVDISA	RDEVFLAG
	RDEVFTR	RDEVNICL	RDEVOWN	RDEVPSUP	RDEVSER	RDEVSIZE	RDEVSTAT	RDEVSYS	RDEVTYPC	RDEVTYPE	RDEVUSER	RDEV333V	RUNUSER
	R0	R1	R10	R11	R12	R13	R14	R15	R2	R3	R4	R5	R6
	R7	R8	R9	SAVEAREA	SAVEREGS	SAVERETN	SAVER11	SAVER2	SAVEWRK1	SAVEWRK2	SAVEWRK6	SAVEWRK8	SAVEWRK9
	START	SYSLOCS	SYSVIRT	TEMPSAVE	TIMEDISP	TRQBIRA	TRQBLOK	TRQBSIZE	TRQEUSER	TYPBSC	TYP1052	TYP2305	TYP3330
	UDBFBLOK	UDBFSIZE	UDBFVADD	UDEVADD	UDEVBLOK	UDEVDED	UDEVDISP	UDEVFTR	UDEVLINK	UDEVLKDV	UDEVLKID	UDEVLONG	UDEVMODE
	UDEVSIZE	UDEVSTAT	UDEVTDSK	UDEVTYPC	UDEVTYPE	UDEVVSER	UDIRBLOK	UDIRDISP	UDIRPASS	UDIRUSER	UMACACC	UMACACCT	UMACAFF
	UMACBLOK	UMACBMX	UMACCDEL	UMACCLA	UMACCLEV	UMACCORE	UMACCPU	UMACDIST	UMACDVCT	UMACECOP	UMACES	UMACFFON	UMACIPL
	UMACISAM	UMACLDEL	UMACLEND	UMACNSVC	UMACOPT	UMACOPT2	UMACPRIR	UMACPUID	UMACRT	UMACVROP	VCHADD	VCHBLOK	VCHSIZE
	VCUADD	VCUBLOK	VCUSIZE	VDEVADD	VDEVBLOK	VDEVFLG2	VDEVREAL	VDEVSIZE	VIRTUAL	VMACCOUN	VMACNT	VMACOUNT	VMAFF
	VMAPTIME	VMBLOK	VMBSIZE	VMCF	VMCFREAD	VMCFWAIT	VMCHCNT	VMCHSTRT	VMCLEVEL	VMCPTIME	VMCPUID	VMCUCNT	VMCUSTRT
	VMDELAY	VMDFTPNT	VMDISC	VMDIST	VMDVCNT	VMDVSTRT	VMECEXT	VMESTAT	VMFEMX	VMFSTAT	VMFVTMR	VMISAM	VMKILL
	VMLOGON	VMMACCON	VMMCODE	VMMCPAST	VMMCPENV	VMMCR6	VMMFE	VMMICRO	VMMICSVC	VMMIMSG	VMMLEVEL	VMMLINED	VMMLVL2
	VMMSGON	VMMSVC	VMMTEXT	VMMVTMR	VMM360	VMNPWOCL	VMOSTAT	VMPNT	VMPSTAT	VMPSW	VMPSWDCT	VMQSTAT	VMREAL
	VMRON	VMRSTAT	VMSEG	VMSIZE	VMSLEEP	VMSTOR	VMSYSCP	VMTCDEL	VMTERM	VMTESCP	VMTIMER	VMTLDEL	VMTLEND
	VMTLEVEL	VMTMOUTQ	VMTON	VMTRMID	VMTRQBLK	VMTTIME	VMUPRIOR	VMUSER	VMVCR0	VMVIRCF	VMVTERM	VMVTIME	VMV370R
	VMWNGON	VRALOC	WAIT	ZEROES									
DMKLOH	AFRET	APSTAT1	APUOPER	AQCNWT	ASYSLC	BLANKS	CLASSPEC	CLASTERM	DMKCQRFI	DMKCVTAB	DMKCVTBD	DMKCVTBH	DMKCVTDT
	DMKFRET	DMKLOKSW	DMKQCNWT	DMKSCNRD	DMKSYSCK	DMKSYSDT	DMKSYSDW	DMKSYSLG	DMKSYSLW	DMKSYSMU	DMKSYSNM	DMKSYSTI	DMKSYSTM
	F1	LOCK	NORET	OPERATOR	PSA	RDEVBLOK	RDEVTYPC	RDEVTYPE	R0	R1	R10	R11	R12
	R13	R14	R15	R2	R3	R4	R5	R6	R7	R8	R9	SAVEAREA	SAVEREGS

MODULE	EXTERNAL REFERENCES (LABELS AND MODULES)												
	SAVER2	SAVEWRK1	SAVEWRK8	SAVEWRK9	SYSLOCS	TIMEDISP	TYPBSC	UDBFSIZE	VMBLOK	VMOSTAT	VMSYSOP	VMTERM	VMTIMEON
	VMTRMID	VMUSER											
DMKLOK	ADSPCH	AFREE	APSTAT1	APSTAT3	APSTAT4	APUOPER	ASYSVM	CODE	CPCREG0	CPEXADD	CPEXBLOK	CPEXPROC	CPEXREGS
	CPEXR1	CPEXR11	CPEXR14	CPEXSIZE	CPLOKFL	CPSYSLK	C0	DMKCVTAB	DMKDSPCH	DMKFREE	DMKSTKSW	DMKVMASW	EMSMASK
	F1	LASTUSER	LOCK	LOKSAVE	LOKSAV2	LPUADDR	MFAMASK	OFF	PREFIXA	PREFIXB	PSA	R0	R1
	R10	R11	R12	R14	R15	R2	R3	R4	R5	R6	R9	SVMNOUPD	SVMSTAY
	SVMUNLOK	TIMEDISP	TRACCURR	TRACEND	TRACFLG2	TRACPROC	TRACSTRT	TRAC12	TRCLCK	VMBLOK	VMDEFSTK	VMLOCK	VMLOCKER
	VMOSTAT	VMPEND	VMSHR	VMSHRPRC	XTNDLOCK								
DMKMCC	ACORETBL	AFREE	AFRET	APTRAN	APTRLK	AQCNWT	ASYSVM	AUTGO	BLANKS	BRING	CC	CLASTAPE	CORCP
	CORFLAG	CORTABLE	CPCREG8	CPEXADD	CPEXBLOK	CPEXR0	CPEXR11	CPEXR12	CPEXSIZE	CPUMODEL	C1	C8	DASDCL
	DEFER	DMKCVTAB	DMKCVTDB	DMKCVTHB	DMKDSPNP	DMKERMSG	DMKFREE	DMKFRET	DMKMCDIN	DMKMCDLI	DMKMCDSE	DMKMCDST	DMKMCDTI
	DMKMIACC	DMKMIADL	DMKMIAMU	DMKMIARO	DMKMNIDK	DMKMNISP	DMKMNIST	DMKMNITH	DMKMONMI	DMKMONTI	DMKPGTVG	DMKPRGC8	DMKPRGMC
	DMKPRGMI	DMKPRGTI	DMKPTRAN	DMKQCNWT	DMKRSPMN	DMKSCHRT	DMKSCHST	DMKSCNFD	DMKSCNRU	DMKSTKCP	DMKSYSAT	DMKSYSEN	FFS
	F0	F1	F10	F2	F3	F4	F5	F8	IOBCAW	IOBLOK	IOEMISC	IOBSIZE	IOBUSER
	LOCK	MNBHDLEN	MNCHSIZE	MONAIOB	MONARDB	MONATRB	MONBUFAC	MONBUFAV	MONBUF1	MONCHPTR	MONCOM	MONCRSLT	MONCURBF
	MONDVLST	MONDVNUM	MONFLAG1	MONFLAG3	MONIOBF	MONIOSLT	MONNEXT	MONSIZE	MONTINT	MONUSER	MONUTRB	MON1BUF	NORET
	PAGECUR	PAGEND	PERFCL	PREFIXA	PREFIXB	PSA	RDEVBLOK	RDEVDED	RDEVDISA	RDEVFLAG	RDEVSTAT	RDEVSYS	RDEVTYPC
	RDEVUSER	R0	R1	R10	R11	R12	R13	R14	R15	R2	R3	R4	R5
	R6	R7	R8	R9	SAVEAREA	SAVEREGS	SAVER12	SAVEWRK1	SAVEWRK3	SCHEDCL	SILI	SPOOLED	SPROFCL
	SYSTEM	TRACCURR	TRACEFLG	TRACSTRT	TRQBIRA	TRQBLOK	TRQBSIZE	TRQBTOD	TRQBUSER	TRQBVAL	USERCL	VMBLOK	VMSEG
	VMUSER												
DMKMCD	ADSPCH	AFREE	AFRET	APSTAT1	APUOPER	AQCNWT	AUTGO	AUTOSPL	BLANKS	CFSTOP	CPCREG8	CPEXADD	CPEXBLOK
	CPEXREGS	CPEXSIZE	C8	DEFINTVL	DMKCFCSC	DMKCVTBH	DMKCVTDB	DMKCVTHB	DMKDSPCH	DMKENTSK	DMKENTUT	DMKERMSG	DMKFREE
	DMKFRET	DMKMIAMU	DMKMNISH	DMKMNIST	DMKPRGC8	DMKPRGMC	DMKPRGTI	DMKQCNWT	DMKSCHRT	DMKSCNFD	DMKSCNRU	DMKSTKOP	DMKSYSAT
	DMKSYSMX	DMKSYSTE	DMKSYSTS	ERROR	F0	F1	F2	F3	F4	F5	F60	F7	F8
	LOCK	MONCHPTR	MONCOM	MONFLAG1	MONFLAG3	MONIOBF	MONSLET	MCNUSER	MONUTRE	NORET	PREFIXB	PSA	RANGE
	R0	R1	R10	R11	R12	R13	R14	R15	R2	R3	R4	R5	R6
	R7	R8	R9	SAVEAREA	SAVEREGS	SAVER12	SAVEWRK1	SAVEWRK3	SAVEWRK4	SAVEWRK6	SPOOLED	START	TODATE
	TOOBIG	TRACEFLG	TRQBSIZE	VMBLOK	VMUSER	X40FFS	ZEROES						
DMKMCH	ACORETBL	ADSPCH	AEXTSP	AFREE	ALARM	ALOKVM	AMCHAREA	APSTAT1	APSTAT2	APSTAT4	APUOPER	AQCNWT	ASYSVM
	AVMREAL	BLANKS	CODE	CORDISA	CORFLAG	CORIOLCK	CORPGPNT	CORSWPNT	CORTABLE	CPCREG0	CPCREG8	CPEXADD	CPEXBLOK
	CPEXREGS	CPEXSIZE	CPID	CPMCHLK	CPMCHSE	CPPTLBR	CPSTATUS	CPUID	CPUVERSN	CPWAIT	CRBIT	C0	C1
	C13	C3	C7	DMKCFMBK	DMKCFPRR	DMKCVTDB	DMKDSPCH	DMKDSPRU	DMKERMSG	DMKFREE	DMKIOEMC	DMKLOKDF	DMKLOKSY
	DMKMCTPT	DMKMCTST	DMKOPRWT	DMKPGSPO	DMKPTRPT	DMKQCNWT	DMKSCNFD	DMKSTKOP	EXDCCF	EXDCNO	EXDRESVD	FFS	F2
	F255	F3	F6	F8	INTMC	INTRC	IPUADDR	IPUADDRX	LOCK	LPUADDR	MCCPUID	MCFXDLOG	MCHAREA
	MCHCPEX	MCHEK	MCHFIX	MCHFLAG0	MCHFLAG1	MCHFLAG3	MCHFLAG4	MCHFLAG5	MCHFLAG6	MCHFLAG7	MCHFSAR	MCHLEN	MCHLEN1
	MCHMODEL	MCHPDAR1	MCHPDAR6	MCHPDAR7	MCHP1IDE	MCHP1IKE	MCHP1SDE	MCHP1SKE	MCHP6CEA	MCHREC	MCHRESEV	MCH0HDWR	MCH0QUIT
	MCH0SFTR	MCH0TERM	MCH0USAD	MCH1BUFF	MCH1COST	MCH1GERR	MCH1MAIN	MCH1PROC	MCH1TODC	MCH3DATA	MCH3INTE	MCH3PROT	MCH3SOLD
	MCH4BURE	MCH4REPA	MCH5IFSA	MCH7EXIT	MCH7IOEM	MCH7OPSW	MCH7PURG	MCH7RSRE	MCH7SMCR	MCH7SUP	MCH7SYST	MCH7VEQR	MCH7VRTM
	MCNPSW	MCOLDPW	MCOPSW	MCPROGID	MCREC	MCRECORD	MCRECTYP	MICBLOK	MICVTMR	MODEL135	MODEL145	MODEL155	MODEL165
	MOD3031	MOD3033	MOD4331	NORET	OFF	OPERATOR	PAGCORE	PAGINVAL	PREFIXA	PREFIXB	PROBMODE	PSA	QUANTUMR
	RECMODE	RECOVRPT	RUNUSER	R0	R1	R10	R11	R12	R13	R14	R15	R2	R3

MODULE	EXTERNAL REFERENCES (LABELS AND MODULES)												
	R4	R5	R6	R7	R8	R9	SAVEAREA	SAVEREGS	SIGSTART	SIGSTOP	START	STOP	SWPCHG1
	SWPCHG2	SWPFLAG	SWPKEY1	SWPKEY2	SWTCH	TIMEDISP	TIMER	TRACCURR	TRACEND	TRACFLG1	TRACPROC	TRACSTRT	TRAC04
	TRANMODE	TRCMCH	TYPE	VMBLOK	VMESTAT	VMEXTCM	VMEXWAIT	VMFPRS	VMGPRS	VMINVPAG	VMKILL	VMLOCK	VMMADDR
	VMMCR6	VMMVTMR	VMOSTAT	VMPSW	VMRSTAT	VMSEG	VMTIMER	VMTMOUTQ	VMUSER	WAITEND	Y0	Y2	Y4
	Y6	ZEROES											
DMKMCT	ADSPCH	AEXTSP	AFREE	ALARM	APSTAT1	APSTAT4	APUOPER	AQCNWT	ASYSVM	AVMREAL	CPAPRPND	CPEXADD	CPEXBLOK
	CPEXREGS	CPEXSIZE	CPID	CPSTATUS	CPSUPER	CPTERMLK	CPWAIT	DMKCFMBK	DMKCFPRR	DMKCPUUP	DMKDSPCH	DMKFREE	DMKLOKDF
	DMKLOKDS	DMKLOKFR	DMKLOKPS	DMKLOKRL	DMKLOKSY	DMKLOKTR	DMKOPRWT	DMKPGSPO	DMKQCNWT	DMKSTKMP	DUMPSAVE	EMSPEND	EMSPQUI
	F255	F3	IPUADDR	IPUADDRX	LOCK	LPUADDR	MCHAREA	MCHFLAG1	MCHFLAG7	MCH1TODC	MCH7OPSW	MFASAVE	NORET
	OFF	OPERATOR	POFFLINE	PREFIXA	PREFIXB	PRIORITY	PROCIO	PSA	RESET	RSRTNPSW	RUNUSER	R0	R1
	R10	R11	R12	R14	R15	R2	R3	R4	R6	R9	SIGAPR	SIGREST	SIGSSS
	SIGSTOP	SIGXC	START	STOP	SWTCH	TIMEDISP	TYPE	VMAFF	VMAFFON	VMBLOK	VMEXWAIT	VMKILL	VMLOGOFF
	VMLOGON	VMOSTAT	VMPNT	VMRSTAT	VMUSER	XCPEND	ZEROES						
DMKMIA	ADDSFB	AFREE	AFRET	APSTAT1	APTRAN	APTRLK	APUOPER	AQCNWT	ARSPRD	ASYSOP	ASYSVM	AUTOSPL	BLANKS
	BRING	CFSTOP	CHGSFB	CLCMD	CLSUS	CPCREG8	CPEXADD	CPEXBLOK	CPEXR0	CPEXR10	CPEXR11	CPEXR12	CPEXSIZE
	C1	C8	DEFER	DMKCKSPL	DMKCVTBD	DMKCVTDT	DMKDSPAC	DMKDSPBC	DMKDSPCC	DMKDSPCK	DMKDSPIT	DMKDSPNP	DMKDSPPT
	DMKDSPRC	DMKENTFI	DMKERMSG	DMKFREE	DMKFRENP	DMKFRET	DMKHVCDI	DMKIOSCT	DMKLOKCT	DMKLOKDS	DMKLOKFR	DMKLOKRL	DMKLOKSW
	DMKLOKSY	DMKLOKTR	DMKMCCCL	DMKMNIDK	DMKMNIDS	DMKMNISH	DMKPAGCC	DMKPAGPS	DMKPGTSG	DMKPGTVG	DMKPGTVR	DMKPRGCT	DMKPRGC8
	DMKPRGMC	DMKPRVCD	DMKPRVCE	DMKPRVCH	DMKPRVCP	DMKPRVCS	DMKPRVCT	DMKPRVDI	DMKPRVEK	DMKPRVEP	DMKPRVIK	DMKPRVIP	DMKPRVLC
	DMKPRVLP	DMKPRVLR	DMKPRVMN	DMKPRVMO	DMKPRVMS	DMKPRVNC	DMKPRVPB	DMKPRVPE	DMKPRVPT	DMKPRVRR	DMKPRVTC	DMKPRVTE	DMKPSANX
	DMKPTRAN	DMKPTRCS	DMKPTRFC	DMKPTRFF	DMKPTRFN	DMKPTRF0	DMKPTRPR	DMKPTRRC	DMKPTRRF	DMKPTRSC	DMKPTRSS	DMKPTRSW	DMKQCNWT
	DMKRPAGT	DMKRPAPT	DMKRSPID	DMKRSPMN	DMKSCHCT	DMKSCHN1	DMKSCHN2	DMKSCHPU	DMKSCNAU	DMKSCNFD	DMKSPLDL	DMKSPLSP	DMKSTKCP
	DMKSYSAT	DMKSYSBF	DMKSYSCL	DMKSYSEN	DMKSYSND	DMKSYSNM	DMKSYSUR	DMKVSICI	DMKVSICT	DMKVSICW	DMKVSIHD	DMKVSIHI	DMKVSISF
	DMKVSISI	DMKVSITC	DMKVSITI	EXHAUST	F0	F1	F2	F3	F4	F5	IPLPSW	LOCK	MNBHDLEN
	MONAIOB	MONBUFAC	MONBUFAV	MONBUFIO	MONBUF1	MONCOM	MONCURBF	MCNDAS	MONDASA	MONDASB	MONEX	MONFLAG1	MONFLAG2
	MONFLAG3	MONIOBF	MONNEXT	MONSFB	MONSPLCT	MONUSER	MON1BUF	NCRET	OPERATOR	OPNSFB	PAGEND	PERFCL	PGREAD
	PGWRITE	PREFIXA	PREFIXB	PSA	PSASVCCT	RDRCHN	R0	R1	R10	R11	R12	R13	R14
	R15	R2	R3	R4	R5	R6	R7	R8	R9	SAVEAREA	SAVEREGS	SAVER1	SAVER2
	SAVEWRK1	SAVEWRK3	SFBCLAS	SFBCOPY	SFBDATE	SFBDIST	SFBFILID	SFBFLAG2	SFBFNAME	SFBFTYPE	SFBLAST	SFBLOK	SFBMON
	SFBOR1G	SFBPNT	SFBRECNO	SFBSIZE	SFBSTART	SFBTIME	SFBTYPE	SFBUSER	SPLINK	SPNXTPAG	SPOOLED	SPPREPAG	SPRECNUM
	SUSPEND	SYSTEM	TIMEDISP	TRACEFLG	TRAP	TYP2540P	UNFIN	VMBLOK	VMSEG	VMUSER	ZEROES		
DMKMID	AFREE	ALARM	APSTAT1	APUOPER	AQCNWT	ASYSOP	ASYSVM	AUTGO	CPEXADD	CPEXBLOK	CPEXR0	CPEXR11	CPEXR12
	CPEXSIZE	DATE	DMKCVTDT	DMKDMPDT	DMKDMPTD	DMKENTKC	DMKERMSG	DMKFREE	DMKLOKSW	DMKMNIST	DMKPRGMC	DMKQCNWT	DMKSCHST
	DMKSTKCP	DMKSYSAT	DMKSYSDW	DMKSYSTE	DMKSYSTI	DMKSYSTS	LOCK	NORET	PREFIXB	PSA	R0	R1	R10
	R11	R12	R13	R14	R15	R2	R3	R4	R5	R6	R7	R8	R9
	SAVEAREA	SAVEREGS	SAVER11	SAVEWRK2	TEMPSAVE	TIMEDISP	TODATE	TRQBLOK	TRQBVAL	VMBLOK	VMMLEVEL	VMMSGON	VMPNT
	ZEROES												
DMKMNI	ACORETBL	ADSPCH	AFREE	AFRET	APSTAT1	APTRLK	APUOPER	AQCNWT	ARIOCH	ARIOCT	ARIOCU	ARIODV	ASYSVM
	AUTGO	AUTOSPL	BLANKS	CC	CFSTOP	CLASDASD	CLASTAPE	CLSUS	CORCP	CORFLAG	CORTABLE	CPCREG8	CPEXADD
	CPEXBLOK	CPEXR0	CPEXR12	CPEXSIZE	CPUID	C1	C8	DASDCL	DEFINTVL	DMKCPEID	DMKCPEND	DMKCVTAB	DMKCVTBD
	DMKCVTDT	DMKDSPCH	DMKDSPNP	DMKENTBS	DMKENTEC	DMKENTES	DMKENTET	DMKENTSC	DMKENTST	DMKENTTB	DMKENTTE	DMKENTTI	DMKENTUT
	DMKERMSG	DMKFREE	DMKFREHI	DMKFRELO	DMKFRET	DMKIOSQR	DMKMIACC	DMKMONPR	DMKMCN00	DMKMON40	DMKPGTVR	DMKPRGC8	DMKPRGMC

MODULE	EXTERNAL REFERENCES (LABELS AND MODULES)												
	DMKPRGMI	DMKPRGTI	DMKPTRLK	DMKPTRUL	DMKQCNWT	DMKSCHRT	DMKSCHST	DMKSCNFD	DMKSTKCP	DMKSYSAT	DMKSYSCL	DMKSYSMX	DMKSYSRM
	DMKSYSRV	DMKSYSTE	DMKSYSTS	DMKSYSUR	DMKUDRFU	ERROR	FFS	F1	F2	F4095	F5	F8	IOBCAW
	IOBFATAL	IOBFLAG	IOBIOER	IOBIRA	IOBLOK	IOBMISC	IOBMISC2	IOBSIZE	IOBSTAT	IOERSIZE	LOCK	LPUADDR	LPUADDRX
	MNCHSIZE	MNCLDAST	MNCLPERF	MNCLUSER	MNCODASH	MNCOSYS	MNCOTH	MNCOTT	MNCOUSER	MNDEVLEN	MN097	MN097APL	MN097CPL
	MN097CPU	MN097CR8	MN097DAT	MN097DPA	MN097FSS	MN097LEN	MN097LEV	MN097NUC	MN097TIM	MN097TTS	MN097UID	MN097VR	MN098
	MN098LEN	MN098UID	MN600ADD	MN600CNT	MN600DEV	MN600DLN	MN600HDR	MN600HLN	MN600MAX	MN600NUM	MN600SER	MN600TY	MONAIOB
	MONARDB	MONATRB	MONBUF1	MONBUF1V	MONCHPTR	MONCLASS	MONCODE	MCNCOM	MONCUREF	MONDVLST	MONDVNUM	MONFLAG1	MONFLAG2
	MONFLAG3	MONIOBF	MONNEXT	MONSFB	MONSIZE	MONSPLCT	MONSUSCT	MONTIINT	MONUSER	MONUTRB	NORET	PAGECUR	PAGEND
	PAGENXT	PERFCL	PREFIXA	PREFIXB	PROCIO	PSA	RCHADD	RCHBLOK	RCHCUTBL	RCUADD	RCUBLOK	RCUDVTBL	RCUPRIME
	RCUSUB	RCUTYPE	RDEVADD	RDEVBLOK	RDEVCUA	RDEVDISA	RDEVFLAG	RDEVIOCT	RDEVSER	RDEVSTAT	RDEVSYS	RDEVTYPC	R0
	R1	R10	R11	R12	R13	R14	R15	R2	R3	R4	R5	R6	R7
	R8	R9	SAVEAREA	SAVEREGS	SAVER2	SFBFILID	SFBLOK	SFCOLED	STOP	SUSPEND	TODATE	TRQBIRA	TRQBLOK
	TRQBSIZE	TRQBTOD	TRQBUSER	TRQBVAL	TRUN	USERCL	VMBLOK	VMSEG	VMUSER	ZEROES			
DMKMON	ADSPCH	AFREE	AFRET	ALOCBLOK	ALOCMAX	ALOCNTMP	ALOCUSED	APSTAT1	APTRAN	APTRLK	APUOPER	ARIODV	ASYSVM
	ATMRSN	BRING	CFSTOP	CLSUS	CONADDR	CONCNT	CONTASK	CPCREG8	CPEXADD	CPEXBLOK	CPEXR0	CPEXSIZE	CUE
	C1	C8	DASDCL	DE	DEFER	DMKCVTAB	DMKDSPCH	DMKDSPNP	DMKENT62	DMKERMSG	DMKFREE	DMKFRENP	DMKFREST
	DMKFRET	DMKIOSNM	DMKIOSQR	DMKMIA	DMKMIACC	DMKMIAWO	DMKMIAX1	DMKMIAX2	DMKMIAY1	DMKMIAY2	DMKMNIFI	DMKMNITR	DMKPRGC8
	DMKPRGMC	DMKPRGTI	DMKPTRAN	DMKPTRUL	DMKSCHAL	DMKSCHN1	DMKSCHPU	DMKSCHQ1	DMKSCHST	DMKSCHW1	DMKSCHW2	DMKSTKCP	DMKSYSAT
	DMKSYSMX	DMKSYSNM	DMKSYSOC	DMKSYSOW	ERROR	F0	F1	F2	F3	F4	F4096	F8	IDLEWAIT
	IOBCAW	IOBCSW	IOBCYL	IOBFATAL	IOBFLAG	IOBIOER	IOBLOK	IOBMISC	IOBMISC2	IOBSIZE	IOBSTAT	IOERSIZE	IONTWAIT
	IPLPSW	LOCK	MNCLDAST	MNCLPERF	MNCLSYS	MNCLUSER	MNCODA	MNCODAS	MNCOSUS	MNCOSYS	MNCOUSER	MNDEVLEN	MNHCLASS
	MNHCODE	MNHDR	MNHDRLEN	MNHRECSZ	MNHTOD	MN000	MN000ATT	MN000EXT	MN000INT	MN000ISD	MN000LEN	MN000PPA	MN000PPC
	MN000PRB	MN000PSI	MN000Q1E	MN000Q2E	MN000WID	MN000WIO	MN000WPG	MN001	MN001LEN	MN001NXR	MN001PRB	MN001WID	MN001WIO
	MN001WPG	MN099	MN099CNT	MN099LEN	MN099TOD	MN10X	MN10XADD	MN10XLEN	MN10XUID	MN10YCNT	MN10YIO	MN10YLEN	MN20X
	MN20XNPP	MN20XPRC	MN20XQNM	MN20XQ1E	MN20XQ1N	MN20XQ2E	MN20XQ2N	MN20XSWS	MN20XUID	MN20XWSS	MN20YTTI	MN20YVTI	MN202APR
	MN202CRD	MN202IOC	MN202LEN	MN202LIN	MN202LPR	MN202PGR	MN202PNC	MN202PRI	MN202PST	MN202REF	MN202RES	MN203LEN	MN204LEN
	MN204PRI	MN4RSV1	MN400	MN400CRD	MN400INT	MN400IOC	MN400LEN	MN400LIN	MN400LPR	MN400PDK	MN400PDR	MN400PGR	MN400PGW
	MN400PNC	MN400PST	MN400QLV	MN400RES	MN400RST	MN400TTI	MN400UID	MN400UPR	MN400VTI	MN400WSS	MN500	MN500INS	MN500LEN
	MN500OVH	MN500UID	MN500VAD	MN600ADD	MN600CNT	MN600DEV	MN600DLN	MN600HDR	MN600HLN	MN600NUM	MN600SER	MN600TY	MN602DLN
	MN602HLN	MN700	MN700ADD	MN700CCY	MN700CYL	MN700DIR	MN700LEN	MN700QCH	MN700QCU	MN700QDV	MN700UID	MN802CLN	MN802CNT
	MN802CTR	MN802DEV	MN802DLN	MN802NAU	MN802NPP	MN802NUM	MN802PGR	MN802PGW	MN802PRB	MN802WID	MN802WIO	MN802WPG	MONAIOB
	MONARDB	MONBUFAC	MONBUFAV	MONBUFIO	MONBUF1	MONCHPTR	MONCLASS	MONCLOCK	MONCODE	MONCOM	MONCRSLT	MONCURBF	MONDVLST
	MONDVNUM	MONFLAG1	MONFLAG2	MONFLAG3	MONIOBF	MONIOSLT	MONLSTBK	MCNNEXT	MONREGS	MONSACT	MONSAVE1	MONSAVE2	MONSLMT
	MONSPLCT	MONSUSCK	MONSUSCT	MONSYSVM	MONTIINT	MONUSER	MONUTRB	MON1BUF	PAGEND	PAGEWAIT	PERFCL	PGREAD	PGWRITE
	PREFIXA	PREFIXB	PROBTIME	PROCIO	PROPSW	PSA	Q1DROP	RCHADD	RCHBLOK	RCHQCNT	RCUADD	RCUBLOK	RCUCHA
	RCUPRIME	RCUQCNT	RCUSUB	RCUTYPE	RDEVADD	RDEVALLN	RDEVBLOK	RDEVCUA	RDEVCYL	RDEVFLAG	RDEVIOCT	RDEVQCNT	RDEVSER
	RDEVSKUP	RDEVTYPC	RDEVTYPE	R0	R1	R10	R11	R12	R13	R14	R15	R2	R3
	R4	R5	R6	R7	R8	R9	SAVEAREA	SAVEREGS	SAVER1	SAVER5	SAVEWRK1	SPOOLED	SPROFCL
	SUSPEND	SYSTEM	TRACPROC	TRAP	TRQBLOK	TRQBTOD	TRQBVAL	TRUN	UC	UE	USERCL	VMAEX	VMBLOK
	VMCRDS	VMEPRIOR	VMINST	VMIOCNT	VMLINS	VMLOGON	VMLSTPRC	VMPAGES	VMPDISK	VMPDRUM	VMPGREAD	VMPGRINQ	VMPGWRIT
	VMPNCH	VMPNT	VMPSTAT	VMPSW	VMQLEVEL	VMQPRIOR	VMQ1	VMRDINQ	VMRSTAT	VMSEG	VMSTEALS	VMTERM	VMTIME
	VMUPRIOR	VMUSER	VMVTIME	VMWSPROJ	ZEROES								
DMKMSG	AFREE	AFRET	ALARM	APSTAT1	APUOPER	AQCNWT	ASYSOP	ASYSVM	BLANKS	BUFFER	BUFNXT	DMKCVTDB	DMKCVTDT
	DMKERMSG	DMKFREE	DMKFRET	DMKLOKSW	DMKQCNRD	DMKQCNWT	DMKSCNAU	DMKSCNFD	DMKVMCFC	F1	F2	F3	LOCK

MODULE	EXTERNAL REFERENCES (LABELS AND MODULES)												
	NORET	NOTIME	NOTRESP	PRIORITY	PSA	R0	R1	R10	R11	R12	R13	R14	R15
	R2	R3	R4	R5	R7	R8	R9	SAVEAREA	SAVEREGS	SAVER11	SAVER2	SAVEWRK1	SAVEWRK2
	SAVEWRK3	SAVEWRK4	SAVEWRK6	SAVEWRK8	TIMEDISP	VMBLOK	VMCLASSA	VMCLASSB	VMCLEVEL	VMCMEFLG	VMCMFUNC	VMCMHDR	VMCMLEN
	VMCMLENA	VMCMMID	VMCMUSE	VMCMUSER	VMCMVADA	VMCPSENX	VMDISC	VMKILL	VMLOGOFF	VMMLEVEL	VMMLINED	VMMSGON	VMMTEXT
	VMOSTAT	VMPNT	VMRSTAT	VMSMSGON	VMSPMFLG	VMSPMON	VMUSER	VMWNGON	XRIGHT16				
DMKMSW	AFREE	AFRET	ALARM	APSTAT1	APUOPER	AQCNWT	ASYSOP	CCC	CDC	CLASDASD	DMKCVTBH	DMKFREE	DMKFRET
	DMKLOKSW	DMKQCNRD	DMKQCNWT	DMKSCNRN	EDIT	F10	F20	F4	F6	F8	F9	IFCC	INTREQ
	IOBLOK	IOBRADD	IOERACT	IOERADR	IOERBLOK	IOERCNCL	IOERCSW	IOERDASD	IOERDATA	IOERDEC	IOERETRY	IOERFLG1	IOERIGN
	IOERIGNR	IOERIND3	IOERIND4	IOERINFO	IOERLEN	IOERNUM	IOERPEND	ICERSTRT	LOCK	NORET	NOTIME	OPERATOR	PSA
	RDEVBLOK	RDEVDED	RDEVIOER	RDEVSTAT	RDEVTYPC	RDEVTYPE	R0	R1	R10	R11	R12	R13	R14
	R15	R2	R3	R4	R5	R6	R7	R8	R9	SAVEAREA	SAVEREGS	SAVER0	SAVER11
	TIMEDISP	TYP3340	TYP3350	UCASE	VMBLOK	VMDISC	VMOSTAT	VMTERM	VMUSER	ZEROES			
DMKNEM	CVTEXIT	R0	R1	R11	R12	R13	R15	R2	R3	R4	R5	SAVEAREA	SAVEREGS
	SAVER0												
DMKNES	AFREE	AFRET	APSTAT1	APUOPER	AQCNWT	ARIOCU	ARIODV	ASYSVM	BLANKS	CACTLTR	CDISPLY	CLASSPEC	CLASTERM
	CONCCW3	CONDATA	CONSYSR	CONTASK	CSWLMEP	CSWLNCP	CTRMLTR	DMKCVTBH	DMKCVTDB	DMKCVTHB	DMKERMSG	DMKFREE	DMKFRET
	DMKIOESR	DMKLOKSW	DMKQCNCL	DMKQCNTO	DMKQCNWT	DMKRGBEN	DMKRICRN	DMKRNHND	DMKRNHTR	DMKSCNFD	DMKSCNRD	DMKSCNRU	FFS
	F1	F255	F3	F4	F4095	LOCK	NICBLOK	NICCIBM	NICDISA	NICENAB	NICEPAD	NICEPMD	NICFLAG
	NICLBSC	NICLINE	NICLTRC	NICPSUP	NICQPNT	NICSESN	NICSIZE	NICSTAT	NICSWEP	NICTYPE	NICUSER	NORET	PROCIO
	PSA	RCHBLOK	RCHCUTBL	RCUBLOK	RCUDISA	RCUDVTBL	RCUSTAT	RDEVADD	RDEVEASE	RDEVBLOK	RDEVCON	RDEVCTRS	RDEVCUA
	RDEVDED	RDEVDISA	RDEVDISB	RDEVENAB	RDEVEPDV	RDEVEPLN	RDEVEPMD	RDEVFLAG	RDEVIRM	RDEVLNCP	RDEVMAX	RDEVMDL	RDEVNICL
	RDEVNRDY	RDEVPDLY	RDEVPTTC	RDEVRCVY	RDEVRSVD	RDEVSADN	RDEVSLOW	RDEVSTAT	RDEVTBTU	RDEVTCTL	RDEVTMCD	RDEVTYPC	RDEVTYPE
	RDEVUSC8	RDEVUSER	RDEVWAIT	R0	R1	R10	R11	R12	R13	R14	R15	R2	R3
	R4	R5	R6	R7	R8	R9	SAVEAREA	SAVEREGS	SAVER11	SAVER2	SAVER9	SAVEWRK1	SAVEWRK2
	SAVEWRK3	SAVEWRK4	SAVEWRK5	SAVEWRK7	SAVEWRK8	SAVEWRK9	SVMSTAY	TIMEDISP	TYPESC	TYPTTY	TYPUNDEF	TYP2700	TYP3705
	VMBLOK	VMOSTAT	VMUSER	VMVIRCF									
DMKNET	AFREE	AFRET	APSTAT1	AQCNWT	ARIODV	ASYSVM	BLANKS	CACTLIN	CDCTLIN	CLASSPEC	CLASTERM	CONCCW3	CONSYSR
	CONTACT	CRESIMD	DEVICE	DMKCVTBH	DMKCVTHB	DMKERMSG	DMKFREE	DMKFRET	DMKIOESR	DMKNESDS	DMKNESEP	DMKNESHD	DMKNESPL
	DMKNESTR	DMKNESWN	DMKNLDR	DMKNLEMP	DMKQCNWT	DMKRGBEN	DMKRICRN	DMKRNHND	DMKSCNFD	DMKSCNRD	DMKSCNRU	F255	F3
	F4	F4095	F60	F8	LOCK	NICBLOK	NICCIBM	NICDISA	NICDISB	NICENAB	NICEPAD	NICEPMD	NICFLAG
	NICGRAF	NICLBSC	NICLGRP	NICLINE	NICNAME	NICRSPL	NICSESN	NICSIZE	NICSTAT	NICTELE	NICTERM	NICTYPE	NICUSER
	NORET	PROCIO	PSA	RDEVAUTO	RDEVBASE	RDEVBLOK	RDEVCTRS	RDEVDED	RDEVDISA	RDEVDISB	RDEVENAB	RDEVFLAG	RDEVLNCP
	RDEVMAX	RDEVNICL	RDEVNRDY	RDEVRSVD	RDEVSTAT	RDEVTYPC	RDEVUSER	R0	R1	R10	R11	R12	R13
	R14	R15	R2	R3	R4	R5	R6	R7	R8	R9	SAVEAREA	SAVEREGS	SAVER2
	SAVER9	SAVEWRK1	SAVEWRK2	SAVEWRK3	SAVEWRK4	SAVEWRK5	SAVEWRK7	SAVEWRK8	SAVEWRK9	TEMPSAVE	TYP3705	VMBLOK	VMCLASSA
	VMCLASSB	VMCLASSC	VMCLASSD	VMCLASSE	VMCLASSF	VMCLASSG	VMCLEVEL	VMOSTAT	VMSTKO	VMUSER	VMVIRCF	ZEROES	
DMKNLD	ABORT	ADSPCH	AFREE	AFRET	APSTAT1	APTRLK	APUOPER	AQCNWT	ASYSVM	ATTN	BLANKS	BRING	CC
	CCPARM	CCPENTRY	CCPMAXID	CCPNAME	CCPPSIZE	CCPRESID	CCPRSTAT	CCPRSTEP	CCPRSTYP	CCPSIZE	CCPTNCP	CCPTPEP	CCPTYPE
	CDC	CE	CLASSPEC	CUE	C1	DE	DEFER	DMKCVTBH	DMKCVTHB	DMKDSPCH	DMKERMSG	DMKFREE	DMKFRET
	DMKIOSQR	DMKLOKSW	DMKPGTVG	DMKPGTVR	DMKPTRAN	DMKPTRUL	DMKQCNCL	DMKQCNRD	DMKQCNTO	DMKQCNWT	DMKRNHIN	DMKRNTBL	DMKRPAGT
	DMKSCNFD	DMKSCNRD	DMKSCNRU	DMKSCNVS	DMKSCNVU	DMKSTKIO	DMKVDREL	EDIT	ERRMSG	FFS	FTRTYP1	F1	F256

MODULE	EXTERNAL REFERENCES (LABELS AND MODULES)												
	F3	F4096	F8	IL	INTREQ	IOBBPNT	IOBCAW	IOBCC1	IOBCC3	IOBCP	IOBCSW	IOBFLAG	IOBFPNT
	IOBIOER	IOBIRA	IOBLINK	IOBLOK	IOBMISC	IOBMISC2	IOBRADD	IOBRCAW	IOBRCNT	IOBRSTRT	IOBSIZE	IOBSPEC	IOBSTAT
	IOBTIO	IOBUNSL	IOBUSER	IOERBLOK	IOERDATA	IOERETN	IOEREXT	IOERSIZE	IPLREQ	LOCK	NCPNAME	NCPPAGCT	NCPPNT
	NCPSTART	NCPTBL	NCPVOL	NICBLOK	NICCIBM	NICEPAD	NICEPMD	NICFLAG	NICNAME	NICPSUP	NICSIZE	NICSTAT	NICSWEP
	NICTERM	NICTYPE	NICUSER	NOAUTO	NORET	NOTRESP	OPERATOR	PROCIO	PSA	RCUBLOK	RCUCHAOF	RCUDISA	RCUDVTBL
	RCUSTAT	RDEVADD	RDEVAIOB	RDEVATT	RDEVBASE	RDEVBLOK	RDEVCODE	RDEVCUA	RDEVDED	RDEVDISA	RDEVENAB	RDEVEPDV	RDEVEPLN
	RDEVEPMD	RDEVFIOB	RDEVFLAG	RDEVFTR	RDEVIRM	RDEVLCEP	RDEVLNCP	RDEVMAX	RDEVMDL	RDEVNCP	RDEVNICL	RDEVNRDY	RDEVOWN
	RDEVPTTC	RDEVRCVY	RDEVRSVD	RDEVSTAT	RDEVSTA2	RDEVTFLG	RDEVTMCD	RDEVTYPC	RDEVTYPE	RDEVUSER	R0	R1	R10
	R11	R12	R13	R14	R15	R2	R3	R4	R5	R6	R7	R8	R9
	SAVEAREA	SAVEREGS	SAVER11	SAVER2	SAVEWRK1	SAVEWRK2	SAVEWRK3	SAVEWRK4	SAVEWRK5	SAVEWRK6	SAVEWRK7	SAVEWRK8	SAVEWRK9
	SILI	SM	SVMSTAY	SYSTEM	TEMPSAVE	TIMEDISP	TYPBSC	TYPIBM1	TYPUNDEF	TYP2314	TYP3330	TYP3350	TYP3705
	UC	UCASE	VCUBLOK	VCUDVTBL	VDEVADD	VDEVBLOK	VDEVDIAL	VDEVFLAG	VMBLOK	VMSEG	VMUSER	X40FFS	
DMKNLE	ABORT	ADDSFB	ADSPCH	AFREE	AFRET	APSTAT1	APTRLK	AQCNWT	ARSPRD	ASYSVM	ATTN	BLANKS	BRING
	CC	CDC	CLASSPEC	CUE	C1	DE	DEFER	DMKCKSPL	DMKCVTBH	DMKCVTDT	DMKCVTHB	DMKDSPCH	DMKERMSG
	DMKFREE	DMKFRET	DMKIOSQR	DMKPGTCG	DMKPGTSD	DMKPGTVG	DMKPGTVR	DMKPTRAN	DMKPTRUL	DMKQCNRD	DMKQCNWT	DMKRNHIN	DMKRPAPT
	DMKRSPID	DMKSCNFD	DMKSCNRD	DMKSCNRU	DMKSTKIO	DMKSYSDU	EDIT	ERRMSG	FTRTYP1	F0	F1	F256	F3
	F4	F4096	F5	F8	IL	INTREQ	IOBCAW	IOBCC1	IOBCC3	IOBCP	IOBCSW	IOBFLAG	IOBIOER
	IOBIRA	IOBLINK	IOBLOK	IOBMISC	IOBMISC2	IOBRADD	IOBRCAW	IOBRCNT	IOBRSTRT	IOBSIZE	IOBSPEC	IOBSTAT	IOBTIO
	IOBUNSL	IOBUSER	IOERBLOK	IOERDATA	IOERETN	IOEREXT	IOERSIZE	IPLREQ	LOCK	NOAUTO	NORET	OPERATOR	PROCIO
	PSA	RDEVAUTO	RDEVBLOK	RDEVDED	RDEVDISA	RDEVFLAG	RDEVFTR	RDEVMDL	RDEVNRDY	RDEVRCVY	RDEVRSVD	RDEVSTAT	RDEVTYPC
	RDEVTYPE	RDEVUSER	RDRCHN	R0	R1	R10	R11	R12	R13	R14	R15	R2	R3
	R4	R5	R6	R7	R8	R9	SAVEAREA	SAVEREGS	SAVER2	SAVEWRK1	SAVEWRK2	SAVEWRK3	SAVEWRK4
	SAVEWRK5	SAVEWRK6	SAVEWRK7	SAVEWRK8	SAVEWRK9	SFBCLAS	SFBCOPY	SFBDATE	SFBDIST	SFBDUMP	SFBFILID	SFBFLAG	SFBFNAME
	SFBFTYPE	SFBLAST	SFBLOK	SFBORIG	SFBPNT	SFBRECNO	SFBRECSZ	SFBSIZE	SFBSTART	SFBTIME	SFBTYPE	SFBUSER	SILI
	SM	SYSTEM	TYPPRT	TYP2314	TYP3330	TYP3350	TYP3705	UC	UCASE	VMBLOK	VMSEG	VMUSER	X40FFS
DMKNMT	BUFFER	ERROR	FREELOWE	FSTFMODE	FSTFNAME	NOTEXT	NUCON	R0	R1	R12	R14	R15	R2
	R3	R4	R5	R6	R7	R8	R9	TEXT					
DMKOPR	ALARM	CAW	CC	CD	CLASGRAF	CPUID	CPUVERSN	CSW	DMKRIOCN	DMKRIODV	EUA	FFS	IC
	LOCK	NOAUTO	PSA	RDEVBLOK	RDEVCORD	RDEVGRTY	RDEVTYPC	RDEVTYPE	R0	R1	R10	R14	R15
	R2	R3	R4	R5	R8	SBA	SF	SILI	TYP3066	UC	XRIGHT16		
DMKPAG	ACORETBL	ADSPCH	AFREE	AFRET	ALOKSP	APSTAT1	APUOPER	ARIODV	ASYSVM	CC	CE	CORTABLE	CPEXADD
	CPEXBLOK	CPEXBPNT	CPEXFPNT	CPEXMISC	CPEXR0	CPEXR11	CPEXR5	CPEXR7	CUE	DE	DMKCVTAB	DMKCVTBH	DMKDSPCH
	DMKFREE	DMKFRET	DMKIOSQR	DMKMCHST	DMKPTRRQ	DMKPTRWQ	DMKSCNRD	DMKSTKCP	DMKSTKIO	DMKSTKMP	DMKSTKOP	DMKSYSOW	FTR70MB
	F1	F2	F3	F4	F5	F8	IL	IOBBPNT	IOBCAW	IOBCC3	IOBCP	IOBCSW	IOBCYL
	IOBFATAL	IOBFLAG	IOBFPNT	IOBIRA	IOBLINK	IOBLOK	IOBMINI	IOBMISC	IOBPAG	IOBRADD	IOBSIZE	IOBSTAT	IOBUSER
	LASTUSER	LOCK	LPUADDR	OWNDLIST	OWNDRDEV	PAGELOAD	PAGERATE	PAGEWAIT	PCI	PREFIXA	PREFIXB	PROCIO	PSA
	RDEVBLOK	RDEVFTR	RDEVIOBL	RDEVMDL	RDEVTYPE	R0	R1	R10	R11	R12	R13	R14	R15
	R2	R3	R4	R5	R6	R7	R8	R9	SILI	SKIP	SWPCODE	SWPCYL	SWPDPAGE
	SWPFLAG	SWPRECMP	SWPTRANS	TIMEDISP	TYPE	TYP2305	TYP2314	TYP3330	TYP3340	TYP3350	VMBLOK	VMDSTAT	VMINQ
	VMLOCK	VMQLEVEL	VMQ1	XTNDLOCK									
DMKPER	VMBLOK	VMPEND	VMPERPND	VMTRCTL	VMTRPER								

MODULE	EXTERNAL REFERENCES (LABELS AND MODULES)												
DMKPGS	ACORETBL	ADSPCH	AFREE	AFRET	APSTAT1	APSTAT2	APTRAN	APUOPER	ARIODV	ASYSVM	AVMREAL	CORCFLCK	CORFLAG
	CORFPNT	CORFREE	CORIOLCK	CORPGPNT	CORRSV	CORSHARE	CORSWPNT	CORTABLE	CPEXADD	CPEXBLOK	CPEXR0	CPEXSIZE	CPPTLBR
	C1	DEFER	DMKBLDRL	DMKBLDRT	DMKCVTAB	DMKDSPCH	DMKDSPNP	DMKFREE	DMKFRET	DMKPGTPR	DMKPGTSP	DMKPTRAN	DMKPTRFT
	DMKPTRPW	DMKPTRRC	DMKPTRRS	DMKPTRSC	DMKPTRUL	DMKSTKCP	DMKSYSAP	DMKSYSOW	DMKVMASH	FFS	F0	F1	F15
	F16	F4096	F8	INUSE	KEEPSEGS	LASTUSER	LOCK	MPFEAT	OLDVMSEG	OWNDLIST	OWNDRDEV	PAGACT	PAGBMP
	PAGCORE	PAGREF	PAGSHR	PAGSTMP	PAGTABLE	PAGTONLY	PAGTOT	PAGTSWP	PREFIXB	PROCIO	PSA	RDEVBLOK	RDEVTYPE
	R0	R1	R10	R11	R12	R13	R14	R15	R2	R3	R4	R5	R7
	R8	R9	SAVEAREA	SAVEREGS	SAVER1	SAVER12	SAVER13	SAVER2	SAVER3	SAVEWRK1	SAVEWRK2	SAVEWRK3	SAVEWRK4
	SAVEWRK5	SAVEWRK6	SAVEWRK7	SAVEWRK9	SEGINV	SEGPAGE	SEGTABLE	SHRBPNT	SHRFLAG	SHRFPNT	SHRNAME	SHRNOPRT	SHRSEGCT
	SHRSEGNM	SHRTABLE	SHRTSIZE	SHRUSECT	SWPALLOC	SWPCODE	SWPCYL	SWPFLAG	SWPKEY1	SWPRECMP	SWPSHR	SWPTABLE	SWPTRANS
	SWPVM	TEMPR1	TEMPR2	TREXANSI	TREXIN1	TREXNSI	TREXT	TYP2305	VMABLOK	VMADSTOP	VMAFPNT	VMANAME	VMASIZE
	VMASSIST	VMBLOK	VMDSP	VMDSTAT	VMESTAT	VMINQ	VMINVPAG	VMLOGOFF	VMNSHR	VMOSTAT	VMPAGES	VMPDISK	VMPDRUM
	VMPSTAT	VMRSTAT	VMSEG	VMSHR	VMSHRSYS	VMSIZE	VMSTOR	VMTIMER	VMTREXT				
DMKPGT	ADSPCH	AFREE	AFRET	ALARM	ALOCBLOK	ALOCMAP	ALOCMAX	ALOCUSED	APSTAT1	AQCNWT	ARIODV	ASYSVM	BALRSAVE
	BALR0	BALR1	BALR8	CPEXADD	CPEXBLOK	CPEXR11	CPEXSIZE	CPID	DMKCKP	DMKDSPCH	DMKFREE	DMKFRET	DMKPTRXX
	DMKQCNWT	DMKSTKCP	DMKSYSOW	FFS	FTR70MB	F1	F3	F4	IOBCYL	IOBFPNT	IOBLOK	LOCK	NORET
	OPERATOR	OWNDLIST	OWNDRDEV	PROCIO	PSA	RDEVALLN	RDEVBLCK	RDEVCODE	RDEVCYL	RDEVFIOB	RDEVFLAG	RDEVFTR	RDEVPAGE
	RDEVPNT	RDEVPREF	RDEVRECS	RDEVTYPE	RECBLOK	RECCYL	RECMAP	RECMAX	RECPNT	RECSIZE	RECUSED	R0	R1
	R10	R11	R12	R13	R14	R15	R2	R3	R4	R5	R6	R7	R8
	R9	SWPCYL	SWPDPAGE	SWPFLAG	SWPRECMP	TYP2305	TYP2314	TYP3330	TYP3340	TYP3350	VMBLOK	VMPDISK	VMPDRUM
DMKPRG	ADSPCH	APSTAT1	APTRAN	APUOPER	AQCNWT	BRING	CODE	CPABEND	CPCREG0	CPCREG8	CPEXADD	CPEXBLOK	CPEXPROC
	CPEXREGS	CPSTATUS	CPSUPER	C0	C1	C8	DEFER	DMKCFMBK	DMKDMPDK	DMKDSPA	DMKDSPB	DMKDSPCH	DMKDSPRU
	DMKLOKDF	DMKLOKSY	DMKPERIL	DMKPRVLG	DMKPTRAN	DMKQCNWT	DMKSTKDE	DMKTRCPG	DMKVATPF	DMKVATPX	DMKVATSX	DUMPSAVE	ECBLOK
	ERRMSG	EXTPERAD	EXTPERCD	FFS	INTPR	INTPRL	INTSVCL	LCCK	LPUADDR	MONCLASS	MONCODE	MONREGS	NORET
	PERADD	PERCODE	PREFIXA	PRNPSW	PROBMODE	PROPSW	PSA	QUANTUMR	RUNCR0	RUNUSER	R0	R1	R10
	R11	R12	R13	R14	R15	R2	R3	R4	R5	R6	R7	R8	R9
	START	STOP	SVCNPSW	SVCOPSW	TEMPR12	TEMPR14	TEMPR15	TIMEDISP	TIMER	TRACCURR	TRACEND	TRACFLG1	TRACPROC
	TRACSTRT	TRAC03	TRANMODE	TRCPGM	TREXADD	TREXINTC	TREXINTL	TREXPERA	TREXPERC	TREXPSW	TREXT	TYPE	VFAULT
	VMBLOK	VMCFRUN	VMCFWAIT	VMDFTPNT	VMDSP	VMDSTAT	VMECEXT	VMESTAT	VMEXTCM	VMEXWAIT	VMFPRS	VMGPRS	VMIOPND
	VMIOWAIT	VMOSTAT	VMPAGEX	VMPEND	VMPERCM	VMPERPND	VMPRGIL	VMPRGPND	VMPSTAT	VMPSW	VMRSTAT	VMSEG	VMSHR
	VMSVCPND	VMTMOUTQ	VMTRBRIN	VMTRCTL	VMTREXT	VMTRPER	VMTRPEG	VMV370R	Y0	Y2	Y4	Y6	
DMKPRV	ADSPCH	ADTRANS	APSTAT1	APSTAT2	APTRAN	APUOPER	BRING	CHANID	CLASDASD	CPCREG0	CPEXADD	CPEXBLOK	CPEXREGS
	CPMICON	CPPTLBR	CPSTAT2	CPUID	CPUMCELL	CPUSER	CPUVERSN	C0	C1	C14	C15	C6	DEFER
	DMKDSPA	DMKDSPB	DMKDSPCH	DMKDSPRU	DMKHVCAL	DMKLOKDF	DMKLOKSY	DMKPERIL	DMKPRGSM	DMKPSAFP	DMKPSASP	DMKPTRAN	DMKSCNVU
	DMKSTKDE	DMKTMR	DMKTMRCC	DMKTMRSP	DMKTMRTN	DMKTRCPB	DMKTRCPV	DMKVATAB	DMKVATAT	DMKVATEX	DMKVATLA	DMKVATRN	DMKVSIEX
	DMKVSIVS	ECBLOK	EXTCR0	EXTCR9	EXTMODE	EXTPERAD	EXTSHCR0	FFS	F15	F16	F240	F4	F5
	F6	F60	F7	INTPR	INTPRL	LOCK	LOCKSAV	LPUADDR	MNCLINST	MNCOSIM	PERGPRS	PERSALT	PROBMODE
	PROPSW	PSA	RCHBLOK	RCHSTIDC	RCUBLOK	RCUCHA	RCUPRIME	RCUSUB	RCUTYPE	RDEVBLOK	RDEVCUA	RUNCR0	R0
	R1	R10	R11	R12	R13	R14	R15	R2	R3	R4	R5	R6	R7
	R8	R9	SAVE	SWPFLAG	SWPKEY1	SWPSHR	TEMPSAVE	TRANMODE	TREXCR9	TREXFLAG	TREXINTC	TREXIN1	TREXNDSP
	TREXNSI	TREXPERA	TREXT	TYPE	VCHBLOK	VCHBMX	VCHCUTBL	VCHSEL	VCHTYPE	VCUBLOK	VCUDVTBL	VDEVBLOK	VDEVDED
	VDEVREAL	VDEVSTAT	VDEVTYPC	VMBLOK	VMCPUID	VMCUSTRT	VMDFTPNT	VMDSP	VMDSTAT	VMDVSTRT	VMECEXT	VMESTAT	VMEXTCM

MODULE	EXTERNAL REFERENCES (LABELS AND MODULES)												
	VMEXTPND	VMEXWAIT	VMGPRS	VMINQ	VMINST	VMINVPAG	VMINVSEG	VMIOINT	VMIOLOG	VMIOPND	VMMADDR	VMMCR6	VMMICRO
	VMMPROB	VMNEWCR0	VMPEND	VMPERCM	VMPERPND	VMPGPND	VMPRGIL	VMPSTAT	VMPSW	VMPXINT	VMREAL	VMRSTAT	VMRUN
	VMSEG	VMTRBRIN	VMTRCTL	VMTREXT	VMTRPER	VMTRPRV	VMVCR0	VMVCR14	VMV370R				
DMKPSA	ACORETBL	ACTIVTRQ	ADSPCH	AFREE	ALOKSP	APSTAT1	APUCPER	ASYSOP	ASYSVM	BUSY	CLASGRAF	CLASTERM	CODE
	CORFLAG	CORSHARE	CORTABLE	CPABEND	CPCREG0	CPCREG8	CPEXADD	CPEXBLOK	CPEXREGS	CPEXR11	CPEXSIZE	CPRUN	CPSTATUS
	CPSUPER	CPWAIT	CRESIMD	CSW	CUE	C0	C8	DMKCPE	DMKDMPDK	DMKDSPCH	DMKDSPE	DMKDSPRU	DMKEXTSL
	DMKEXTSP	DMKFREE	DMKFRET	DMKLOKDF	DMKLOKSP	DMKLOKSY	DMKLOKTR	DMKLOKVM	DMKPRVMA	DMKPTRAN	DMKPTRLK	DMKQCNCL	DMKQCNWT
	DMKRIOCC	DMKRIOCH	DMKRIOCT	DMKRIOCU	DMKRIODC	DMKRIODV	DMKRICPR	DMKRIOPU	DMKRIORD	DMKRIOUC	DMKRNHND	DMKRSPAC	DMKRSPPR
	DMKRSPPU	DMKRSPRD	DMKSCHTQ	DMKSCNRD	DMKSTKIO	DMKSTKMP	DMKSVCNS	DMKSYSCS	DMKSYSLC	DMKSYSOP	DMKSYSVM	DMKTMRSN	DMKTMRVT
	DUMPSAVE	EXOPSW	F15	F2	F240	F4095	F60	F8	INTEX	INTEXP	INTKPLIN	LOCK	LPUADDR
	NICBLOK	NICNAME	NICSIZE	NICUSER	NORET	PREFIXA	PREFIXB	PROCIO	PSA	PSARSV6	QUANTUMR	RDEVBASE	RDEVBLOK
	RDEVFLAG	RDEVHIO	RDEVNICL	RDEVTYPC	RDEVUSER	RUNUSER	R0	R1	R10	R11	R12	R14	R15
	R2	R3	R4	R5	R7	R9	SM	START	TEMPSAVE	TIMEDISP	TIMER	TRACCURR	TRACEND
	TRACFLG1	TRACPROC	TRACSTRT	TRAC01	TRCEXT	TRQBBPNT	TRQBFPNT	TRQBLOK	TRQEVAL	TYPE	VMBLOK	VMCPUTMR	VMDISC
	VMDSP	VMDSTAT	VMESTAT	VMEXTCM	VMFPRS	VMGPRS	VMOSTAT	VMPSW	VMQSEND	VMSHR	VMSYSOP	VMTERM	VMTLEVEL
	VMTMOUTQ	VMTMRINT	VMTRMID	WAITEND	XPAGNUM	X2048BND	Y0	Y2	Y4	Y6	ZEROES		
DMKPTR	ACORETBL	ADSPCH	AEXTSP	AFREE	AFRET	APSTAT1	APSTAT2	APTRAN	APUOPER	AQCNWT	ARIODV	ASYSVM	AVMREAL
	BALRSAVE	BALR0	BALR2	BRING	CORBPNT	CORCFLCK	CORCP	CORFLAG	CORFLUSH	CORFPNT	CORFREE	CORIOLCK	CORLCNT
	CORPGPNT	CORRSV	CORSHARE	CORSWPNT	CORTABLE	CORVM	CPEXADD	CPEXBLOK	CPEXFPNT	CPEXMISC	CPEXR0	CPEXR11	CPEXR13
	CPEXR2	CPEXR7	CPEXR9	CPEXSIZE	CPPTLBR	C1	DEFER	DMKBLDRT	DMKCVTAB	DMKDSPCH	DMKDSPNP	DMKFREAP	DMKFREE
	DMKFRET	DMKFRETR	DMKLOKDF	DMKPAGIO	DMKPAGQ	DMKPGTPG	DMKPGTPR	DMKQCNWT	DMKSCHDL	DMKSCHN1	DMKSCHN2	DMKSTKCP	DMKSTKMP
	DMKSTKOP	DMKSYSCS	DMKSYSOW	DMKSYSRM	FFS	FREEQ	FREESAVE	F0	F1	F16	F2	F4096	IOERETN
	IPUADDRX	LASTUSER	LOCK	LOCKLIST	LOCKSAV	LPUADDR	MICBLOK	MICVTMR	NEWPAGES	NORET	OWNDLIST	OWNDRDEV	PAGACT
	PAGBMP	PAGCORE	PAGINVAL	PAGREF	PAGSHR	PAGSTMP	PAGTABLE	PAGTONLY	PAGTSWP	PGREAD	PGWRITE	PREFIXA	PREFIXB
	PROCIO	PSA	RDEVBLOK	RDEVTYPE	R0	R1	R10	R11	R12	R13	R14	R15	R2
	R3	R4	R5	R6	R7	R8	R9	SAVE	SAVEAREA	SAVEPROC	SAVEREGS	SAVERETN	SAVER0
	SAVER1	SAVER11	SAVER12	SAVER13	SAVER2	SAVER3	SAVER7	SAVEWRK1	SAVEWRK2	SAVEWRK6	SAVEWRK8	SAVEWRK9	SEGINV
	SEGPAGE	SHRFLAG	SHRNOPRT	SHRSEGCT	SHRTABLE	SIGEMS	SIGEXT	SIGQUI	SIGRES	SIGXC	SWPALLOC	SWPAPP	SWPCHG1
	SWPCHG2	SWPCODE	SWPCYL	SWPDPAGE	SWPFLAG	SWPFLAG2	SWPKEY1	SWPKEY2	SWPRECMP	SWPREF1	SWPREF2	SWPSHR	SWPTABLE
	SWPTRANS	SWPVPAGE	SYSTEM	TEMPR0	TEMPR1	TEMPR2	TIMEDISP	TIMER	TYPE	TYP2305	VFAULT	VMBLOK	VMCPWAIT
	VMDSP	VMDSTAT	VMELIG	VMESTAT	VMFLPAG	VMINQ	VMINVPAG	VMLOCK	VMLOCKER	VMMADDR	VMMCR6	VMMVTMR	VMNDCNT
	VMOSTAT	VMPAGES	VMPGREAD	VMPGRINQ	VMPGWAIT	VMPGWRIT	VMPSTAT	VMRON	VMRPAGE	VMRSTAT	VMSEG	VMSHR	VMSIZE
	VMSTEALS	VMSTOR	VMTIMER	VMTLEVEL	VMTON	VMWCNT	VMXPG	XPAGNUM	XTNDLOCK	ZEROES			
DMKQCN	ADSPCH	AFREE	AFRET	ALARM	APSTAT1	APTRAN	APUOPER	AQCNWT	ASYSOP	BALRSAVE	BALR11	BLANKS	BRING
	CLASGRAF	CLASSPEC	CLASTERM	CONADDR	CONCNT	CONCNTL	CONDATA	CONDWC	CONOUTPT	CONPARM	CONPNT	CONRESP	CONRETN
	CONSPLT	CONSTAT	CONSYNC	CONTASK	CONTSIZE	CONTSKSZ	CONUSER	CPEXADD	CPEXBLOK	CPEXREGS	CPEXR12	CPEXR2	CPEXSIZE
	C1	DEFER	DFRET	DMKCNSIC	DMKCVTAB	DMKCVTBD	DMKCVTBH	DMKCVTDT	DMKDSPCH	DMKFREE	DMKFRET	DMKGRFIC	DMKGRTDS
	DMKLOKSW	DMKPTRAN	DMKRGBIC	DMKRNHIC	DMKSCHDL	DMKSCHRT	DMKSCHST	DMKSCNRD	DMKSCNRN	DMKSTKCP	DMKSYSNM	DMKVSPVP	EDIT
	F1	F4095	F9	INHIBIT	LOCK	LOGDROP	LOGHOLD	MNCLRESP	MNCOERD	MNCOERD	MNCOWRIT	NICAPL	NICBLOK
	NICLLEN	NICSIZE	NICTMCD	NOAUTO	NORET	NOTIME	NOTRESP	OPERATOR	PRIORITY	PROCIO	PSA	RDEVACTV	RDEVAPLP
	RDEVBLOK	RDEVCON	RDEVDROP	RDEVFLAG	RDEVGRTY	RDEVLLEN	RDEVNICL	RDEVSTA2	RDEVTMCD	RDEVTYPC	RDEVTYPE	R0	R1
	R10	R11	R12	R13	R14	R15	R2	R3	R4	R5	R6	R7	R8
	R9	SAVEAREA	SAVEREGS	SAVER0	SAVER1	SAVER11	SAVER2	SAVER3	SAVEWRK1	SAVEWRK2	SAVEWRK3	SAVEWRK4	TEMPSAVE

MODULE	EXTERNAL REFERENCES (LABELS AND MODULES)												
	TIMEDISP	TRQBIRA	TRQBLOK	TRQBSIZE	TRQBUSER	TRQBVAL	TYPBSC	UCASE	VDEVELOK	VDEVCSPL	VDEVFLAG	VDEVSFLG	VDEVTERM
	VMBLOK	VMCF	VMCFREAD	VMCFWAIT	VMCOMND	VMCONBUF	VMCONLN	VMDELAY	VMDISC	VMDVSTRT	VMGENIO	VMKILL	VMLOGOFF
	VMLOGON	VMMCODE	VMMLEVEL	VMMSTMP	VMMTEXT	VMOSTAT	VMQSTAT	VMRBSC	VMRSTAT	VMSEG	VMSYSOP	VMTERM	VMTRMID
	VMUSER	VMVIRCF	VMVTERM	XPAGNUM									
DMKRGA	ADSPCH	AFREE	AFRET	APSTAT1	APUOPER	ASYSVM	ATTR2	ATTR457	ATTR7	BALRSAVE	BLANKS	BRING	BSCAUSER
	BSCBLOK	BSCCNT	BSCCOPY	BSCECCW1	BSCECCW2	BSCENQ	BSCETB	BSCFLAG	BSCFLAG1	BSCFORCE	BSCHALT	BSCIGN	BSCINBID
	BSCINDEX	BSCLOG	BSCOPIED	BSCPA1	BSCPCCW1	BSCPCCW2	BSCPCCW3	BSCPCCW4	BSCRCVD	BSCREAD	BSCREGEN	BSCRESP	BSCRPTR
	BSCRROBN	BSCRSTRT	BSCRVI	BSCSCAN	BSCSCCW1	BSCSCCW2	BSCSCCW3	BSCSEL	BSCSEND	BSCSENSE	BSCSIZE	BSCSIZE1	BSCSPTR
	BSCTMRQ	BSCTSTRQ	BSCUCOPY	BSCUECCW	BUFCNT	BUFFER	BUFINLTH	BUFSIZE	CC	CD	CE	CLASTERM	CONACTV
	CONADDR	CONCCW1	CONCCW2	CONCCW3	CONCCW4	CONCNT	CONCNTL	CONDATA	CONDCNT	CONESCP	CONLABEL	CONPARM	CONPNT
	CONRESP	CONRETN	CONSTAT	CONTASK	CONTSIZE	CONTSKSZ	CONUSER	CPEXADD	CPEXBLOK	CPEXR0	CPEXSIZE	C1	DE
	DEFER	DMKBLDVM	DMKCFMAT	DMKCFMBK	DMKCFMEN	DMKCNSED	DMKCVTAB	DMKCVTED	DMKCVTEH	DMKCVTDB	DMKCVTHB	DMKDSPCH	DMKERMSG
	DMKFREE	DMKFRET	DMKIOERN	DMKIOSQR	DMKLOKSW	DMKPTRAN	DMKQCNCL	DMKQCNET	DMKQCNTO	DMKQCNWT	DMKRGBEN	DMKRGBIC	DMKRGBMT
	DMKRGBSN	DMKSCHRT	DMKSCHST	DMKSCNRD	DMKSCNRU	DMKSTKMP	DMKTBLGR	DMKTBLUP	DMKTEMTI	DMKTBMZI	EDIT	ETX	EUA
	FTRDIAL	F0	F1	F2	F20	F3	F4	F4095	F5	F6	F8	IC	INHIBIT
	IOBCAW	IOBCC3	IOBCP	IOBCSW	IOBFATAL	IOBFLAG	IOBIOER	IOBIRA	IOBLINK	IOBLOK	IOBMISC	IOBMISC2	IOBRADD
	IOBRCNT	IOBRSTRT	IOBSIZE	IOBSPEC	IOBSTAT	IOBUNSL	IOBUSER	ICERBLOK	IOEREXT	IOERSIZE	LOCK	LOGDROP	LOGHOLD
	MNCLRESP	MNCOERD	NICALRM	NICAPL	NICATRB	NICBLOK	NICCARD	NICCORD	NICCPNA	NICDIAG	NICDISA	NICDISB	NICENAB
	NICFLAG	NICFMT	NICHOLD	NICLGRP	NICMORE	NICNAME	NICNTRL	NICPOLL	NICPROCN	NICQPNT	NICREAD	NICRSPL	NICRUNN
	NICSELT	NICSIO	NICSIZE	NICSTAT	NICTABP	NICTERM	NICTEXT	NICTMCD	NICTRQ	NICTYPE	NICUSER	NIC3275	NORET
	NOTEXT	NOTIME	NOTRESP	PREFIXA	PROCIO	PSA	RA	RDEVBLOK	RDEVESC	RDEVCON	RDEVDISA	RDEVDISB	RDEVENAB
	RDEVFLAG	RDEVFTR	RDEVMAX	RDEVNICL	RDEVNRDY	RDEVPDLY	RDEVRSVD	RDEVSTAT	RDEVTYPC	RDEVTYPE	RDEVWAII	R0	R1
	R10	R11	R12	R13	R14	R15	R2	R3	R4	R5	R6	R7	R8
	R9	SAVEAREA	SAVER2	SBA	SF	SILI	SVMSTAY	SVMUNLOK	SYSTEM	TABEND	TEMPR0	TEMPR7	TIMEDISP
	TRQBIRA	TRQBLOK	TRQBSIZE	TRQBUSER	TRQBVAL	TYPBSC	UCASE	UE	VCONCTL	VCONRESZ	VCONRBUF	VCONRCNT	VDEVBLOK
	VDEVCON	VMBLOK	VMCF	VMCFWAIT	VMDVSTRT	VMGENIO	VMLOGOFF	VMLOGON	VMMCPENV	VMMLEVEL	VMMLINED	VMOSTAT	VMPA2APL
	VMPFUNC	VMPXINT	VMQSTAT	VMRSTAT	VMSEG	VMTERM	VMTLEND	VMVTERM	WCC3	WCC4	WCC6	XINTBLOK	XINTCODE
	XINTNEXT	XINTSIZE	XINTSORT	XTNDLOCK									
DMKRGB	ADSPCH	AFREE	AFRET	ALARM	APSTAT1	APUOPER	ASYSVM	ATTR2	ATTR457	ATTR7	BALRSAVE	BRING	BSCAUSER
	BSCBLOK	BSCFLAG	BSCFLAG1	BSCHALT	BSCINBID	BSCLINE	BSCPCCW1	BSCPCCW2	BSCPCCW4	BSCRCVD	BSCREAD	BSCRESP	BSCRROBN
	BSCSCAN	BSCSCCW1	BSCSCCW2	BSCSCCW3	BSCSEL	BSCSIZE	BSCSIZE1	BSCSIZE2	BSCSPTR	BUFINLTH	CC	CD	CONADDR
	CONCCW1	CONCCW2	CONCCW3	CONCCW4	CONCNT	CONCNTL	CONDATA	CONDWC	CONESCP	CONLABEL	CONOUTPT	CONPARM	CONPNT
	CONRESP	CONRETN	CONSTAT	CONSYNC	CONTASK	CONTSIZE	CONTSKSZ	CONUSER	CPEXADD	CPEXBLOK	CPEXSIZE	C1	DEFER
	DMKBOXBX	DMKDSPCH	DMKFREE	DMKFRET	DMKIOSHA	DMKIOSQR	DMKLOKSW	DMKPTRAN	DMKQCNET	DMKRGAIN	DMKSCHRT	DMKSTKCP	DMKTBLGR
	DMKTBLRG	DMKTBMTO	DMKTBMZO	ETX	FTRDIAL	F1	F255	F256	F4	F4095	IC	INHIBIT	IOBCAW
	IOBCP	IOBFLAG	IOBIOER	IOBIRA	IOBLOK	IOBMISC	IOBMISC2	IOBRCNT	IOBRSTRT	IOBSIZE	IOBSPEC	IOBSTAT	IOBUSER
	IOERBLOK	IOEREXT	IOERSIZE	LOCK	LOGDROP	LOGHOLD	NICALRM	NICAPL	NICATRB	NICBLOK	NICCORD	NICDIAG	NICDISA
	NICDISB	NICFLAG	NICFMT	NICHOLD	NICMORE	NICNTRL	NICPOLL	NICPROCN	NICQPNT	NICREAD	NICRUNN	NICSELT	NICSIO
	NICSIZE	NICSTAT	NICTEXT	NICTMCD	NICTRQ	NICUSER	PRIORITY	PROCIO	PSA	QUEUE	RA	RDEVAIOB	RDEVBLOK
	RDEVBSC	RDEVCON	RDEVDED	RDEVDISA	RDEVDISB	RDEVFLAG	RDEVFTR	RDEVMAX	RDEVNICL	RDEVNRDY	RDEVRSVD	RDEVSTAT	RDEVWAII
	R0	R1	R10	R11	R12	R13	R14	R15	R2	R3	R4	R5	R6
	R7	R8	R9	SAVEAREA	SAVEREGS	SAVER2	SBA	SF	SILI	SVMSTAY	SYSTEM	TEMPR3	TIMEDISP
	TRQBLOK	VMBLOK	VMGENIO	VMLOGOFF	VMRSTAT	VMSEG	VMTLEND	VMTRMID	WCC3	WCC5	WCC56	WCC6	

MODULE	EXTERNAL REFERENCES (LABELS AND MODULES)												
DMKRND	BUFFER	ERROR	FFS	FSCBFN	INPUT	ON	R0	R1	R10	R11	R12	R13	R14
	R15	R2	R3	R4	R6	R7	SAVEAR	TEXT					
DMKRNH	ABORT	ADSPCH	AFREE	AFRET	ALARM	APSTAT1	APUOPER	AQCNWT	ASYSVM	ATTN	BALRSAVE	BLANKS	BUSOUT
	BUSY	CACTDEV	CACTLIN	CACTLTR	CC	CCDESMD	CDC	CHC	CKPBITS	CKPBKSZ	CKPBLOK	CKPNAME	CKPRMAX
	CKPSIZE	CLASSPEC	CMDREJ	CNTLBTU	CODE	CONACTV	CONADDR	CONCCW1	CONCCW2	CONCCW3	CONCNT	CONCNTL	CONCOMND
	CONDATA	CONDCNT	CONDEST	CONESCP	CONEXTR	CONFLAG	CONOUTPT	CONPARM	CONPNT	CONRESP	CONRETN	CONRTAG	CONRTRY
	CONSPLT	CONSRID	CONSTAT	CONSYNC	CONSYSR	CONTACT	CONTASK	CONTCMD	CONTSIZE	CONTSKSZ	CONUSER	CPEXADD	CPEXBLOK
	CPEXSIZE	CRESCND	CRESERL	CRESIMD	CSETDSM	CTRMLTR	DE	DFRET	DISCEOC	DISCNCT	DMKBLDVM	DMKCFMAT	DMKCFMBK
	DMKCNSED	DMKCPVAE	DMKCVTBH	DMKCVTDT	DMKDSPCH	DMKERMSG	DMKFREE	DMKFRET	DMKICERN	DMKIOSQR	DMKLOKSW	DMKNLDR	DMKNLEMP
	DMKQCNCL	DMKQCNET	DMKQCNTO	DMKQCNWT	DMKRIORN	DMKSCNAU	DMKSCNRU	DMKSTKCP	DMKVSPRT	EDIT	ERRMSG	ERROR	F1
	F16	F256	F4	F4095	F60	F8	IL	INHIBIT	INTREQ	IOBCAW	IOBCC1	IOBCC3	IOBCP
	IOBCSW	IOBFLAG	IOBIOER	IOBIRA	IOBLINK	IOBLOK	IOBMISC	IOBMISC2	IOBRADD	IOBRCAW	IOBRCNT	IOBRSTRT	IOBSIZE
	IOBSPEC	IOBSTAT	IOBUNSL	IOBUSER	IOERBLOK	IOERDATA	IOEREXT	IOERSIZE	IPLREQ	LOCK	LOGDROP	LOGHOLD	NICATOF
	NICATTN	NICBLOK	NICCIBM	NICDED	NICDISA	NICDISB	NICENAB	NICEPMD	NICERLK	NICFLAG	NICLINE	NICLTRC	NICMTA
	NICNAME	NICNTRL	NICPSUP	NICQPNT	NICRCNT	NICSESN	NICSIZE	NICSTAT	NICTELE	NICTERM	NICTYPE	NICUSER	NOAUTO
	NORET	OPERATOR	PCI	PREFIXA	PRGC	PRIORITY	PROCIC	PRTC	PSA	RDBUFLN	RDBUFNO	RDEVAUTO	RDEVBLOK
	RDEVBUSY	RDEVCKPT	RDEVCON	RDEVDED	RDEVDISA	RDEVFLAG	RDEVLCEP	RDEVLNCP	RDEVMAX	RDEVNCP	RDEVNICL	RDEVNRDY	RDEVRCVY
	RDEVRSVD	RDEVSCED	RDEVSLOW	RDEVSTAT	RDEVTBTU	RDEVTYPC	RDEVWAIT	READBUF	READNRM	R0	R1	R10	R11
	R12	R13	R14	R15	R2	R3	R4	R5	R6	R7	R8	R9	SAVEAREA
	SAVEREGS	SAVER0	SAVER1	SAVER2	SILI	SVMSTAY	SVMUNLOK	SYSTEM	TEMPSAVE	TIMEDISP	TRACCURR	TRACEND	TRACFLG2
	TRACPROC	TRACSTRT	TRAC11	TRCNCP	TYP3705	UC	UCASE	UE	VMBLCK	VMCFWAIT	VMLOGON	VMMCPENV	VMMLEVEL
	VMOSTAT	VMRSTAT	VMSLEEP	VMTRMID	VMUSER	VMVIRCF	WRITBRK	WRITEOT	WRITNRM	XRIGHT16	ZEROES		
DMKRPA	ACORETBL	AFREE	APSTAT2	APTRAN	AVMREAL	BRING	CORCFLCK	CORFLAG	CORFPNT	CORIOLCK	CORPGPNT	CORRSV	CORSWPNT
	CORTABLE	CPEXADD	CPEXBLOK	CPEXFPNT	CPEXR0	CPEXSIZE	CPPTLER	DEFER	DMKFREE	DMKPAGIO	DMKPGSPR	DMKPGTPR	DMKPGTSP
	DMKPTRAN	DMKPTRFT	DMKPTRUL	DMKPTRWQ	DMKSCHDL	DMKVMI	FFS	F1	F4	IOERETN	LOCK	MICBLOK	MICVTMR
	PAGCORE	PAGINVAL	PAGREF	PREFIXB	PSA	R0	R1	R11	R12	R13	R14	R15	R2
	R3	R5	R7	R9	SAVEAREA	SAVEREGS	SAVER1	SAVER2	SAVEWRK1	SAVEWRK2	SWPCHG1	SWPCHG2	SWPCYL
	SWPFLAG	SWPRECMP	SWPSHR	SWPTRANS	SYSTEM	VMBLOK	VMESTAT	VMEXWAIT	VMINST	VMINVPAG	VMMADDR	VMMCR6	VMMVTMR
	VMNORUN	VMPAGES	VMPGWAIT	VMRSTAT	VMTIMER	VMWCNT	XPAGNUM	ZEROES					
DMKRSE	ACNTBACK	ACNTBLOK	ADSPCH	AFREE	AFRET	APSTAT1	ATTN	BUSOUT	CC	CCC	CDC	CE	CHC
	CLASURI	CLASURO	CMDREJ	CUE	DATACHK	DE	DMKCVTBD	DMKCVTBH	DMKDSPCH	DMKERMSG	DMKFREE	DMKFRET	DMKIOESD
	DMKIOEST	DMKIOSQR	DMKMSWR	DMKRSPPR	DMKRSP83	EQCHK	FTREXTSN	F1	F15	F3	F4	F7	F8
	IFCC	INTREQ	IOBCAW	IOBCC1	IOBCC3	IOBCSW	IOBERP	IOBFATAL	IOBFLAG	IOBIOER	IOBIRA	IOBLOK	IOBMISC
	IOBMISC2	IOBRADD	IOBRCAW	IOBRCNT	IOBRSTRT	IOBSIZE	IOBSPEC	IOBSTAT	IOBTIO	IOBUSER	IOERACT	IOERBLOK	IOERCCRA
	IOERCCRL	IOERCEMD	IOERCSW	IOERDATA	IOERDEPD	IOERDERD	IOERECSW	ICERERP	IOERETRY	IOEREXT	IOERFLG1	IOERFLG2	IOERFLG3
	IOERIGN	IOERIND3	IOERINFO	IOERLEN	IOERNUM	IOERPEND	IOERPNT	IOERREAD	IOERSIZE	IOERSNSZ	IOERXERP	LOCK	PCI
	PRGC	PROCIO	PRTC	PSA	RDEVACNT	RDEVAIOB	RDEVBACK	RDEVBLOK	RDEVBUSY	RDEVDELP	RDEVFLAG	RDEVFTR	RDEVIOER
	RDEVNRDY	RDEVRSTR	RDEVSPL	RDEVSTAT	RDEVTERM	RDEVTYPC	RDEVTYPE	READBUF	R0	R1	R10	R11	R12
	R13	R14	R15	R2	R3	R4	R5	R6	R7	R8	R9	SAVEAREA	SAVEREGS
	SAVEWRK1	SAVEWRK2	SAVEWRK3	SAVEWRK4	SAVEWRK6	SFBFILID	SFBFLAG	SFBLOK	SFBPNT	SFBRECER	SFBSHOLD	SILI	SKIP
	SM	TYPPUN	TYP1403	TYP1443	TYP2501	TYP2520P	TYP2540P	TYP2540R	TYP3203	TYP3211	TYP3505	TYP3800	UC
	VMBLOK	VMCF	VMOSTAT	XOBRCCW1	XOBRCCW2	XOBRCCW3	XOBRCCW4	XOBREXT	XOBRFLAG	XOBRMIS1	XOBRMIS2	XOBRRT1	XOBRRT2
	XOBRRT3	XOBRRT4	XOBRRT5	XOBRRT6	XOBRSIZE	XOBRSTAT	XOBRT1	XOBRT2	XOBRT3	XOBR010	XOBR150	XOBR180	XOBR512

MODULE	EXTERNAL REFERENCES (LABELS AND MODULES)												
	ZEROES												
DMKRSP	ADSPCH	AFREE	AFRET	ALARM	APSTAT1	APTRAN	APTRLK	AQCNWT	ASYSVM	BLANKS	BRING	BUFCNT	BUFFER
	BUFNXT	BUFSIZE	CC	CCC	CDC	CE	CHGSFB	CLASURI	CLASURO	CPEXADD	CPEXBLOK	CPEXSIZE	C1
	DE	DEFER	DMKACOPU	DMKCKSPL	DMKCSOSD	DMKCVTBD	DMKCVTBH	DMKCVTDT	DMKDSPCH	DMKERMSG	DMKFREE	DMKFRET	DMKIOSQR
	DMKOPRWT	DMKPGTSG	DMKPGTVG	DMKPGTVR	DMKPTRAN	DMKQCNWT	DMKRPAGT	DMKRPAPT	DMKRSERR	DMKSCNFD	DMKSCNRD	DMKSCNRN	DMKSCNRU
	DMKSEPSP	DMKSPLCR	DMKSPLDL	DMKSPLOR	DMKSTKCP	DMKSYSOC	DMKSYSCW	DMKSYSRM	DMKSYSTP	DMKSYSWM	DMKTCSCO	DMKTCSET	DMKTMRPT
	DMKUDRFU	F24	F4	F4095	F4096	F8	IFCC	IL	IOBCAW	IOBCC1	IOBCP	IOBCSW	IOBERP
	IOBFATAL	IOBFLAG	IOBIOER	IOBIRA	IOBLOK	IOBMISC	IOBRADD	IOBRCAW	IOBRCNT	IOBRSTRT	IOBSIZE	IOBSPEC	IOBSTAT
	IOBUNSL	IOERBLOK	IOERCCRA	IOERCCRL	IOERCSW	IOERDATA	IOERDEPD	IOERDERD	IOERERP	IOEREXT	IOERFLG1	IOERSIZE	LOCK
	NORET	OPERATOR	PROCIO	PSA	RDEVACNT	RDEVBACK	RDEVBLCK	RDEVBUSY	RDEVCLAS	RDEVDED	RDEVDELP	RDEVDISA	RDEVDRAN
	RDEVFLAG	RDEVIOER	RDEVLOAD	RDEVMAXP	RDEVNRDY	RDEVOVLY	RDEVPURG	RDEVRSTR	RDEVSEP	RDEVSPAC	RDEVSPL	RDEVSTAT	RDEVSTA2
	RDEVTERM	RDEVTYPC	RDEVTYPE	RECBLOK	RECCYL	RECMAP	RECPNT	RECSIZE	RECUSED	RSPBF1IO	RSPBF1VL	RSPBF2IO	RSPBF2VL
	RSPDPAGE	RSPDPAG2	RSPFLAG1	RSPLCTL	RSPMISC	RSPRPAGE	RSPRPAG2	RSPRSTRT	RSPSFBLK	RSPSIZE	RSPVPAGE	RSPVPAG2	R0
	R1	R10	R11	R12	R13	R14	R15	R2	R3	R4	R5	R6	R7
	R8	R9	SAVEAREA	SAVEREGS	SAVER0	SFBCLAS	SFBCOPY	SFBFILID	SFBFLAG	SFBFLAG2	SFBFLASH	SFBFNAME	SFBFTYPE
	SFBINUSE	SFBLAST	SFBLOK	SFBORIG	SFBPNT	SFBRECER	SFBRECNO	SFBRECOK	SFBRECS	SFBREQUE	SFBRSTRT	SFBSHOLD	SFBSIZE
	SFBSTART	SFBTICER	SFBTYPE	SFBUHOLD	SFBUSER	SILI	SKIP	SPLINK	SPNXTPAG	SPPREPAG	SPRECNUM	SPRMISC	SPSIZE
	SYSTEM	TYPPRT	TYPPUN	TYPRDR	TYP2540R	TYP3800	UC	UE	VMBLOK	VMSEG			
DMKSAV	ALARM	CAW	CC	CE	CSW	DE	DMKCKP	DMKCKPLD	DMKCKPRS	DMKCKPST	DMKCKPT	DMKCPICD	DMKCPINT
	DMKCVTBH	DMKOPRWT	DMKSYSNU	DMKSYSRS	DMKSYSTP	DMKSYSTZ	DMKSYSVL	EXNPSW	F1	F2	F3	F4	INTREQ
	INTTIO	IONPSW	IOOPSW	LOCK	MCNPSW	PRNPSW	PSA	PSTARTSV	R0	R1	R10	R11	R12
	R13	R14	R15	R2	R3	R4	R5	R6	R7	R8	R9	SILI	SKIP
	START	TEMPR2	TEMPR4	TEMPSAVE	TYP2305	TYP2314	TYP3330	TYP3340	TYP3350				
DMKSCH	ACTIVTRQ	ADSPCH	AEXTSP	AFREE	AFRET	ALOKSP	APSTAT1	APUOPER	ATMRSN	AVMREAL	BALRSAVE	BALR11	CODE
	CPRUN	CPSTATUS	CPSUPER	CPWAIT	C1	DMKCVTAB	DMKDSPCH	DMKDSPNP	DMKFREE	DMKFRET	DMKLOKDS	DMKLOKRL	DMKLOKTR
	DMKMIDNT	DMKPTRRL	DMKPTRRS	DMKPTRUC	ECBLOK	EXTCPTMR	EXTCPTRQ	FFS	F1	F10	F15	F16	F3
	F8	IDLEWAIT	IONTWAIT	IPUADDRX	LOCK	LOCKSAV	LPUADDR	MNCLSCH	MNCOAEL	MNCOAQ	MNCODQ	PAGELOAD	PAGEWAIT
	PREFIXA	PREFIXB	PROBSTRT	PROBTIME	PROCIO	PSA	Q1DROP	RUNUSER	R0	R1	R10	R11	R12
	R13	R14	R15	R2	R3	R4	R5	R6	R7	R8	R9	SAVE	SIGWAKE
	SIGXC	START	STOP	TEMPR1	TEMPR14	TEMPR15	TEMPR4	TEMPSAVE	TIMEDISP	TIMER	TRACCURR	TRACEND	TRACFLG1
	TRACPROC	TRACSTRT	TRAC08	TRAC09	TRCDROP	TRCSCH	TRQBBPNT	TRQBFPNT	TRQELOK	TRQBQUE	TRQBTOD	TRQBVAL	TYPE
	VMAEX	VMAEXP	VMBLOK	VMCOMP	VMCPUTMR	VMCPWAIT	VMDROP1	VMDSP	VMDSTAT	VMECEXT	VMELIG	VMEPRIOR	VMHIPRI
	VMIDLE	VMINQ	VMIOINT	VMLONGWT	VMLOPRI	VMNORUN	VMPAGES	VMPEND	VMPGREAD	VMPGRINQ	VMPRIDSP	VMPSTAT	VMPSWAIT
	VMPXINT	VMQBPNT	VMQFPNT	VMQLEVEL	VMQPRIOR	VMQSEND	VMQSTAT	VMQ1	VMRDINQ	VMRON	VMRPAGE	VMRPRIOR	VMRSTAT
	VMRUN	VMSEG	VMSTEALS	VMSTKCNT	VMSTMPI	VMSTMPT	VMTIMER	VMTLEVEL	VMTMINQ	VMTMOUTQ	VMTMRINT	VMTODINQ	VMTRQBLK
	VMTSEND	VMTTIME	VMUPRIOR	VMVTIME	VMV370R	VMWSERNG	VMWSPROJ	XINTBLOK	XINTCODE	XINTNEXT	XINTPARM	XINTSIZE	XINTSORT
	ZEROES												
DMKSCN	ARIOCH	ARIOCT	ARIOCU	ARIODC	ARIODV	ASYSVM	BALRSAVE	BALR1	BALR2	BALR3	BALR8	BLANKS	BUFFER
	BUFNXT	CLASDASD	CLASSPEC	CLASTERM	CLASURI	CLASURO	FFS	FTR2311B	FTR2311T	F0	F5	F7	PSA
	RCHADD	RCHBLOK	RCHCUTBL	RCUADD	RCUBLOK	RCUCHA	RCUCHC	RCUDVTBL	RCUPRIME	RCUSUB	RCUTYPE	RDEVADD	RDEVBLOK
	RDEVCUA	RDEVCUB	RDEVDED	RDEVDISA	RDEVFLAG	RDEVLNKS	RDEVMCUT	RDEVSER	RDEVSIZE	RDEVSTAT	RDEVTYPC	R0	R1
	R10	R11	R14	R15	R2	R3	R4	R5	R6	R7	R8	R9	TYPCTCA

MODULE	EXTERNAL REFERENCES (LABELS AND MODULES)												
	TYPIBM1	TYPPRT	TYPPUN	TYPRDR	TYPTELE2	TYP2311	TYP2700	TYP3210	TYP3705	TYP3851	UDEVBLOK	UDEVFTR	UDEVRELN
	VCHADD	VCHBLOK	VCHCUTBL	VCUADD	VCUBLOK	VCUDVTBL	VDEVADD	VDEVBLOK	VDEVDED	VDEVDET	VDEVFLAG	VDEVLINK	VDEVRDO
	VDEVREAL	VDEVRELN	VDEVSIZE	VDEVSTAT	VDEVTYPC	VDEVTYPE	VDEVUSER	VMBLOK	VMCHSTRT	VMCHTBL	VMCUSTRT	VMDVCNT	VMDVSTRT
	VMLOGOFF	VMLOGON	VMPNT	VMRSTAT	VMUSER	ZEROES							
DMKSEP	ADSPCH	AFRET	APSTAT1	APTRAN	APTRLK	ASYSVM	BRING	CC	C1	DEFER	DMKBOXBX	DMKCPEID	DMKCVTBD
	DMKCVTBH	DMKCVTDT	DMKDSPCH	DMKFRET	DMKIOSQR	DMKPGTVG	DMKPGTVR	DMKPTRAN	DMKPTRUL	DMKSCNRD	DMKTCSSP	FILE	IOBCAW
	IOBCSW	IOBFATAL	IOBFLAG	IOBIRA	IOBLINK	IOBLOK	IOBMISC	IOBMISC2	IOBRSTRT	IOBSIZE	IOBSTAT	LOCK	PROCIO
	PSA	RDEVBLOK	RDEVFLAG	RDEVLOAD	RDEVSEP	RDEVTYPE	R0	R1	R10	R11	R12	R13	R14
	R15	R2	R3	R4	R5	R6	R7	R8	R9	SAVEAREA	SAVEREGS	SAVER10	SAVER8
	SAVEWRK1	SAVEWRK2	SAVEWRK5	SAVEWRK7	SAVEWRK8	SAVEWRK9	SFBCLAS	SFBDATE	SFBDIST	SFBFILID	SFBFLAG2	SFBFNAME	SFBLOK
	SFBORIG	SFBRECNO	SFBRSTRT	SFBTIME	SFBUSER	SILI	SKIP	SYSTEM	TYPPUN	TYP3800	UE	VMBLOK	VMSEG
DMKSEV	CCC	CCHCHNL	CCHCMDV	CCHCNTB	CCHCPU	CCHDAV	CCHDI	CCHINTFC	CCHLOG70	CCHREC	CCHSTG	CCHUSV	COMPFES
	COMPSEL	COMPSYS	CSW	FFS	F7	F8	HIOCCH	IFCC	IGBLAME	IGPRGFLG	IGTERMSQ	IGVALIDB	INTERCCH
	IOERBLOK	PSA	RTCODE1	RTCODE2	RTCODE3	RTCODE4	RTCODE5	RTCODE7	R0	R1	R12	R13	R14
	R15	R2	R3	R4	R9	SAVEAREA	SAVEWRK1	SAVEWRK9	TERMSYS	TIOCCH	XRIGHT16		
DMKSIX	CCHCHNL	CCHCMDV	CCHCNTB	CCHCPU	CCHDAV	CCHDI	CCHINTFC	CCHLOG60	CCHREC	CCHSTG	CCHUSV	COMPFES	COMPSEL
	CSW	FFS	F1	F7	F8	HIOCCH	IFCC	IGBLAME	IGPRGFLG	IGTERMSQ	IGVALIDB	IOERBLOK	PSA
	RTCODE1	RTCODE2	RTCODE3	RTCODE4	RTCODE5	R0	R1	R12	R13	R14	R15	R2	R3
	R4	R9	SAVEAREA	SAVEWRK1	SAVEWRK9	TERMSYS	TIOCCH	XRIGHT16					
DMKSNC	APTRAN	APTRLK	ASYSVM	BRING	CCPADDR	CCPARM	CCPNAME	CCPPSIZE	CCPSIZE	C1	DEFER	DMKERMSG	DMKPTRAN
	DMKPTRUL	DMKRNTBL	DMKRPAPT	DMKSCNVS	F1	F256	F4096	ICERETN	LOCK	NCPNAME	NCPPAGCT	NCPPNT	NCPSTART
	NCPTBL	NCPVOL	PSA	RDEVBLOK	RDEVCODE	RDEVFLAG	RDEVOWN	RDEVTYPE	R0	R1	R10	R11	R12
	R13	R14	R15	R2	R3	R4	R5	R6	R7	R8	R9	SAVEAREA	SAVEREGS
	SAVER0	SAVER2	SAVER6	SAVEWRK1	SAVEWRK2	SAVEWRK3	SAVEWRK4	SAVEWRK5	SAVEWRK6	SAVEWRK8	SAVEWRK9	SYSTEM	TYP2314
	TYP3330	TYP3350	VMBLOK	VMSEG									
DMKSPL	ACCTBLOK	ACCTDIST	ACCTUSER	ACORETBL	ADDSFB	ADSPCH	AFREE	AFRET	APSTAT1	APTRAN	APTRLK	APUOPER	AQCNWT
	ARIODV	ARIOPR	ARIOPU	ARSPPR	ARSPPU	ARSPRD	ASYSVM	BLANKS	BRING	CC	CHGSFB	CLASURI	CPEXADD
	CPEXBLOK	CPEXREGS	CPEXSIZE	C1	DF	DEFER	DELSFB	DMKCKSPL	DMKCVTBD	DMKCVTDT	DMKDRDDD	DMKDSPCH	DMKFREE
	DMKFRET	DMKIOSQR	DMKLOKSW	DMKPGTSD	DMKPGTSG	DMKPGTSR	DMKPGTVG	DMKPTRAN	DMKPTRLK	DMKPTRUL	DMKQCNWT	DMKRPAGT	DMKRPAPT
	DMKRSPDL	DMKRSPEX	DMKRSPHQ	DMKRSPID	DMKSCNAU	DMKSTKCP	DMKSTKIO	DMKSYSOC	DMKSYSOW	DMKUDRFU	DMKUDRMD	DMKUDRRV	DMKVIOIN
	FTR70MB	F0	F1	F2	F3	F4	IOBCAW	IOBCP	IOBCSW	IOBCYL	IOBFATAL	IOBFLAG	IOBIRA
	IOBLINK	IOBLOK	IOBMISC2	IOBRADD	IOBSIZE	IOBSTAT	IOBUSER	ICBVADD	LOCK	NORET	NOTRESP	OWNDLIST	OWNDRDEV
	PCHCHN	PROCIO	PRTCHN	PSA	RDEVACNT	RDEVBLOK	RDEVCLAS	RDEVDED	RDEVDISA	RDEVDRAN	RDEVFLAG	RDEVFTR	RDEVSPL
	RDEVSTAT	RDEVTYPE	RDRCHN	READ	RECBLOK	RECPNT	RECSIZE	RSPDPAGE	RSPLCTL	RSPRPAGE	RSPSFBLK	RSPSIZE	RSPVPAGE
	R0	R1	R10	R11	R12	R13	R14	R15	R2	R3	R4	R5	R6
	R7	R8	R9	SAVEAREA	SAVEREGS	SAVER11	SAVER7	SAVER8	SAVER9	SAVEWRK1	SAVEWRK2	SAVEWRK8	SEEK
	SFBCLAS	SFBCOPY	SFBDATE	SFBDIST	SFBDUMP	SFBFILID	SFBFIRST	SFBFLAG	SFBFLAG2	SFBFLASH	SFBFNAME	SFBHOLD	SFBLAST
	SFBLOK	SFBMON	SFBNOHLD	SFBORIG	SFBPNT	SFBPURGE	SFBRECER	SFBRECNO	SFBRECS	SFBRECSZ	SFBREQUE	SFBRSTRT	SFBSHOLD
	SFBSIZE	SFBSTART	SFBSTCPY	SFBTIME	SFBTYPE	SFBUHOLD	SFBUSER	SHQBLOK	SHQSHOLD	SHQUSER	SILI	SKIP	SPCHAR
	SPCMOD	SPCOPYFG	SPENDSIZ	SPFCB	SPFLAG1	SPFLSHC	SPLINK	SPSIZE	SVMSTAT	SVMUNLOK	SYSTEM	TIMEDISP	TYPPRT
	TYPPUN	TYPRDR	TYP1052	TYP2314	TYP3210	TYP3211	TYP3340	TYP3350	UDBFELOK	UDBFSIZE	UDBFVADD	UDIRBLOK	UDIRDISP

MODULE	EXTERNAL REFERENCES (LABELS AND MODULES)												
	UMACBLOK	UMACDIST	VCHADD	VCHBLOK	VCHCUTBL	VCUADD	VCUBLOK	VCUDVTBL	VDEVADD	VDEVBLOK	VDEVCLAS	VDEVCOPY	VDEVCSPL
	VDEVCSW	VDEVDLY	VDEVEXTN	VDEVFLAG	VDEVFOR	VDEVHOLD	VDEVPEND	VDEVSFLG	VDEVSPL	VDEVSTAT	VDEVTYPC	VDEVTYPE	VDEVXFER
	VMACOUNT	VMBLOK	VMCHSTRT	VMCHTBL	VMCUSTRT	VMDIST	VMDVSTRT	VMMLEVEL	VMMSGON	VMSEG	VMUSER	VSPLCTL	VSPSFBLK
	VSPSIZE	VSPVPAGE	VSPXBLOK	VSPXCHAR	VSPXCMOD	VSPXCPYF	VSPXFCB	VSPXFLG1	VSPXFLSH	VSPXOVLY	VSPXTAG	VSPXTGLN	VSPXXUSR
	ZEROES												
DMKSSP	ARIOCH	ARIOCT	ARIOCU	ARIODV	ATTN	BUSY	CAW	CC	CD	CE	CLASDASD	CLASGRAF	CLASTAPE
	CLASTERM	CLASURI	CLASURO	CONPARM	CPUID	CPUVERSN	CSW	CUE	DE	DMKCPINT	DMKCVTBH	DMKCVTHB	DMKRIOCH
	DMKRIOCN	DMKRIOCU	DMKRIODV	DMKRIOPR	DMKRIOPU	DMKRIORD	DMKSYSNU	DMKSYSTP	ERROR	EUA	FTREXTSN	FTRUCS	F4096
	IC	INPUT	IOBCSW	IOBFATAL	IOBRCAW	IONPSW	IOOPSW	ICESW	MCNPSW	PRNPSW	PSA	RA	RCHBLOK
	RCHCUTBL	RCHSIZE	RCUADD	RCUBLOK	RCUCHA	RCUCHB	RCUCHC	RCUCHD	RCUDVTBL	RCUSIZE	RCUTYPE	RDEVADD	RDEVBLOK
	RDEVCLAS	RDEVCUA	RDEVFSEP	RDEVFTR	RDEVIMAG	RDEVMAXP	RDEVSIZE	RDEVTYPC	RDEVTYPE	RDEVXSEP	R0	R1	R10
	R11	R12	R13	R14	R15	R2	R3	R4	R5	R6	R7	R8	R9
	SAVEAREA	SBA	SENSE	SF	SILI	SM	SYSTEM	TIC	TYPPRT	TYPPUN	TYP2314	TYP2540P	TYP2540R
	TYP3066	TYP3210	TYP3277	TYP3340	TYP3350	TYP3800	UC	UE	XRIGHT16				
DMKSSS	ADSPCH	AFREE	AFRET	APTRAN	ARIODC	ARIODV	ATTN	BLANKS	BRING	CLASDASD	CPEXADD	CPEXBLOK	CPEXMISC
	CPEXREGS	CPEXR0	CPEXR1	CPEXR11	CPEXR12	CPEXR3	CPEXSIZE	C1	DEFER	DEMOUNT	DMKCVTBH	DMKDSPCH	DMKERMSG
	DMKFREE	DMKFRET	DMKLNKSS	DMKLOGSS	DMKPTRAN	DMKSCHRT	DMKSCHST	DMKSCNAU	DMKSCNRU	DMKSCNVS	DMKSCNVU	DMKSTKCP	DMKVDAS1
	DMKVDAS2	DMKVIOMK	F1	IOBBPNT	IOBCYL	IOBFLT	IOBFPNT	ICBLOK	IOBRADD	IOBRSV3	IOBSIZE	IOBSTAT	IOBVADD
	LOCK	MOUNT	MSSERR	MSSFLAGS	MSSNEXT	MSSPRES	MSSRSRVD	MSSSER	MSSSIZE	MSSTASK1	MSSTASK2	MSSUSER	MSSVUA
	PSA	PSAMSS	RCHADD	RCHBLOK	RCUADD	RCUBLOK	RCUCHA	RCUPRIME	RCUSUB	RCUTYPE	RDEVADD	RDEVATT	RDEVBLOK
	RDEVCUA	RDEVDED	RDEVDISA	RDEVFIOB	RDEVFLAG	RDEVFTR	RDEVLNKS	RDEVMOUT	RDEVOWN	RDEVSEL	RDEVSER	RDEVSIZE	RDEVSTAT
	RDEVSYS	RDEVTYPC	RDEVTYPE	RDEVUSER	R0	R1	R10	R11	R12	R13	R14	R15	R2
	R3	R4	R5	R6	R7	R8	R9	SAVEAREA	SAVEREGS	SAVERETN	SAVER1	SAVER12	SAVER2
	SAVEWRK1	SAVEWRK2	SAVEWRK3	SAVEWRK4	SAVEWRK5	SAVEWRK6	SAVEWRK7	SAVEWRK8	SYSVIRT	TEMPSAVE	TRQBIRA	TRQBLOK	TRQBSIZE
	TRQBTOD	TRQBUSER	TRQBVAL	TYP3330	UDEVADD	UDEVBLOK	UDEVLINK	UDEVVSER	VCHADD	VCHBLOK	VCHCUINT	VCUADD	VCUBLOK
	VCUDVINT	VDEVADD	VDEVBLOK	VDEVBND	VDEVBUSY	VDEVINTS	VDEVPEND	VDEVRELN	VDEVSTAT	VIRTUAL	VMBLOK	VMGPRS	VMIOINT
	VMIOPND	VMPEND	VMPSW	VMSEG	VMUSER	ZEROES							
DMKSTK	ALOKSP	APSTAT1	APUOPER	CPEXBLOK	CPEXDEFR	CPEXMISC	CPEXPRIO	CPEXPROC	CPEXR11	CPEXTYPE	DMKDSPRQ	DMKLOKDS	F1
	IOBFPNT	IOBLOK	IOBUSER	LOCK	LPUADDR	LPUADDRX	PSA	R0	R1	R10	R11	R12	R14
	R15	R2	R3	R4	R5	R6	SWTHSAVE	TYPE	VMBLOK	VMDEFSTK	VMPEND	VMPRRCT	VMSTKCNT
DMKSVC	ADSPCH	AFREE	AFRET	APAGCP	APSTAT1	APTRAN	APTRLK	APUOPER	AQCNWT	ASYSVM	BRING	CODE	CPABEND
	CPCREG0	CPCREG8	CPEXADD	CPEXBLOK	CPEXLPSW	CPEXREGS	CPEXR12	CPEXSIZE	CPEXTYPE	CPRUN	CPSTATUS	CPSUPER	C0
	C1	C8	DEFER	DFRET	DMKCFMBK	DMKCVTBH	DMKDMPDK	DMKDSPB	DMKDSPCH	DMKDSPRU	DMKFREE	DMKFRET	DMKLOKDF
	DMKLOKSY	DMKPRGRF	DMKPTRAN	DMKPTRUL	DMKQCNWT	DMKSTKDE	DMKSTKSW	DMKTRCIT	DMKTRCPB	DMKTRCSV	DMKVERD	DMKVERO	DUMPSAVE
	ECBLOK	EXTCR8	EXTMASK	EXTMODE	FFS	F4095	F60	INTSVC	INTSVCL	IOERETN	IOMASK	LOCK	LPUADDR
	MCHEK	NORET	PREFIXA	PROBMODE	PSA	PSACPXBP	PSASVCCT	QUANTUM	QUANTUMR	RUNCH0	RUNPSW	RUNUSER	R0
	R1	R10	R11	R12	R13	R14	R15	R2	R3	R4	R5	R6	R7
	R8	SAVEAREA	SAVENEXT	SAVEPROC	SAVERETN	SAVERTN	SAVER0	SAVER12	SAVER13	SAVESIZE	SAVEWRK2	START	STOP
	SVCNPSW	SVCOPSW	SVCREGS	SYSTEM	TIMEDISP	TIMER	TRACCURR	TRACEND	TRACFLG1	TRACPROC	TRACSTRT	TRAC02	TRANMODE
	TRCSVC	TREXIN1	TREXT	TYPE	VMADSTOP	VMBLOK	VMDFTENT	VMECEXT	VMESTAT	VMEXTCM	VMEXWAIT	VMFPRS	VMGPRS
	VMINST	VMMCR6	VMMICSVC	VMMSVC	VMPEND	VMPERPND	VMPSTAT	VMPSW	VMRSTAT	VMSEG	VMTMOUTQ	VMTRBRIN	VMTRCTL
	VMTREXT	VMTRSVC	VMV370R	WAIT	XRIGHT24	Y0	Y2	Y4	Y6				

MODULE	EXTERNAL REFERENCES (LABELS AND MODULES)												
DMKSYM	DMKACO	DMKALG	DMKAPI	DMKATS	DMKBLD	DMKBOX	DMKBSCER	DMKCCHIS	DMKCCHNT	DMKCCHRT	DMKCCH60	DMKCCWSB	DMKCCWTC
	DMKCCWTR	DMKCDB	DMKCDBDC	DMKCDM	DMKCDS	DMKCFC	DMKCFD	DMKCFDAD	DMKCFG	DMKCFH	DMKCFMAT	DMKCFMBK	DMKCFMEN
	DMKCFOEX	DMKCFP	DMKCFPRD	DMKCFSET	DMKCFT	DMKCKPT	DMKCKS	DMKCLK	DMKCNSED	DMKCNSEN	DMKCNSIC	DMKCNSIN	DMKCPB
	DMKCPEID	DMKCPEML	DMKCPEND	DMKCPI	DMKCPS	DMKCPSH	DMKCPU	DMKCPUUP	DMKCPUVY	DMKCPV	DMKCQG	DMKCQP	DMKCQR
	DMKCQY	DMKCSB	DMKCSO	DMKCSP	DMKCSQ	DMKCST	DMKCSU	DMKCSV	DMKCSVTR	DMKDASER	DMKDGDDK	DMKDIA	DMKDIB
	DMKDMPDK	DMKDRD	DMKDSBRD	DMKDSBSD	DMKDSPA	DMKDSPAC	DMKDSPB	DMKDSPBC	DMKDSPCC	DMKDSPCH	DMKDSPE	DMKDSPEC	DMKDSPNP
	DMKDSPQS	DMKDSPRC	DMKDSPRQ	DMKDSPRU	DMKEMA	DMKEMB	DMKEMC	DMKERM	DMKEXTSL	DMKEXTSP	DMKFCB	DMKFREE	DMKFREHI
	DMKFRELG	DMKFRELO	DMKFRELS	DMKFRENP	DMKFRERC	DMKFRERS	DMKFRESV	DMKFRET	DMKFRETR	DMKGIO	DMKGRFEN	DMKGRFIC	DMKGRFIN
	DMKGRFTI	DMKHVCAL	DMKHVCDI	DMKHVD	DMKIOC	DMKIOEFM	DMKIOERR	DMKIOF	DMKIOG	DMKIOSHA	DMKIOSIN	DMKIOSMQ	DMKIOSQR
	DMKIOSQV	DMKISM	DMKLNK	DMKLNKSB	DMKLOC	DMKLOG	DMKLOGOP	DMKLOHON	DMKLOKCT	DMKLOKDF	DMKLOKDS	DMKLOKFR	DMKLOKPS
	DMKLOKRL	DMKLOKSP	DMKLOKSW	DMKLOKSY	DMKLOKTR	DMKLOKVM	DMKMCC	DMKMCHIN	DMKMCHMS	DMKMCHSE	DMKMCHST	DMKMCTMA	DMKMCTPR
	DMKMCTPT	DMKMCTST	DMKMIDNT	DMKMNI	DMKMON	DMKMSG	DMKMSWR	DMKNEM	DMKNES	DMKNET	DMKNLE	DMKOPRWT	DMKPAGIO
	DMKPAGPS	DMKPAGST	DMKPER	DMKPGS	DMKPGSPO	DMKPGSPP	DMKPGTBN	DMKPGTPG	DMKPGTTM	DMKPGTTU	DMKPRGCT	DMKPRGC8	DMKPRGIN
	DMKPRGMC	DMKPRGRF	DMKPRGSM	DMKPRVLG	DMKPRVMC	DMKPSADU	DMKPSAEX	DMKPTRAN	DMKPTRCT	DMKPTRFC	DMKPTRFF	DMKPTRPR	DMKPTRRC
	DMKPTRRQ	DMKPTRSC	DMKPTRSS	DMKPTRWQ	DMKQCNCL	DMKQCNET	DMKQCNRD	DMKQCNSY	DMKQCNTO	DMKQCNWT	DMKRGAIN	DMKRGBEN	DMKRGBIC
	DMKRIOCH	DMKRIOCN	DMKRIOCU	DMKRIODV	DMKRIOPR	DMKRIOPU	DMKRICRD	DMKRIORN	DMKRNHCT	DMKRNHIC	DMKRNHIN	DMKRNHND	DMKRNHTG
	DMKRNHTR	DMKRPAGT	DMKRPAPT	DMKRSE	DMKRSPAC	DMKRSPCV	DMKRSPDL	DMKRSPER	DMKRSPEX	DMKRSPHQ	DMKRSPID	DMKRSPPR	DMKRSPPU
	DMKRSPRD	DMKRSPUR	DMKRSP83	DMKSCHAE	DMKSCHAL	DMKSCHAP	DMKSCHAU	DMKSCHCP	DMKSCHCT	DMKSCHDL	DMKSCHIB	DMKSCHMD	DMKSCHN1
	DMKSCHN2	DMKSCHPB	DMKSCHPD	DMKSCHPG	DMKSCHPU	DMKSCHQ1	DMKSCHQ2	DMKSCHRL	DMKSCHRT	DMKSCHST	DMKSCHTI	DMKSCHTQ	DMKSCHUB
	DMKSCHW1	DMKSCHW2	DMKSCH80	DMKSEP	DMKSEV70	DMKSIX60	DMKSPL	DMKSTKCP	DMKSTKDE	DMKSTKIO	DMKSTKLF	DMKSTKMP	DMKSTKOP
	DMKSVCIN	DMKSYSCS	DMKSYSLC	DMKSYSOC	DMKSYSOP	DMKSYSOW	DMKSYSRM	DMKSYSRS	DMKSYSRV	DMKSYSVL	DMKSYSVM	DMKTAPER	DMKTBL
	DMKTBM	DMKTCS	DMKTDK	DMKTHI	DMKTMRTN	DMKTRA	DMKTRKFP	DMKTRKIN	DMKTRKVA	DMKTRM	DMKUCB	DMKUCC	DMKUCS
	DMKUDR	DMKUDRDS	DMKUNTFR	DMKUNTIS	DMKUNTRN	DMKUNTRS	DMKUSC	DMKVATAB	DMKVATEC	DMKVATEX	DMKVATLA	DMKVATMD	DMKVATPX
	DMKVATRN	DMKVATSX	DMKVCA	DMKVCH	DMKVCNEX	DMKVCNFT	DMKVDA	DMKVDC	DMKVID	DMKVDE	DMKVDR	DMKVER	DMKVIOIN
	DMKVMASH	DMKVMC	DMKVMI	DMKVSICT	DMKVSICW	DMKVSIEX	DMKVSPCO	DMKVSPCP	DMKVSPCR	DMKVSPEX	DMKVSPRT	DMKVSPTO	DMKVSPVP
	DMKVSPWA	DMKVSQPD	DMKWRM	LOCK									
DMKTAP	ADSPCH	AFREE	AFRET	APSTAT1	ASYSVM	CC	CCC	CD	CDC	CHC	CPEXBLOK	CPEXREGS	CPEXR0
	CPEXSIZE	CUE	DE	DMKDSPCH	DMKFREE	DMKFRET	DMKIOEST	DMKIOSQR	DMKMSWR	DMKPTRUL	FFS	FTRRSRL	F1
	F15	F16	F2	F3	F4	F5	F6	F8	IDA	IFCC	IL	IOBCAW	IOBCC3
	IOBCP	IOBCSW	IOBERP	IOBFATAL	IOBFLAG	IOBIOER	IOBIRA	IOBLINK	IOBLCK	IOBMISC	IOBRADD	IOBRCAW	IOBRCNT
	IOBRSTRT	IOBSIZE	IOBSPEC	IOBSTAT	IOBTIO	IOBUNSL	IOBUSER	ICERACT	IOERADR	IOERBLOK	IOERBSR	IOERCAN	IOERCCRA
	IOERCCRL	IOERCLN	IOERCSW	IOERDATA	IOERDW	IOERERG	IOEREXT	ICERFLG1	IOERFLG2	IOERFLG3	IOERFSR	IOERIGNR	IOERIND3
	IOERIND4	IOERINFO	IOERLOC	IOERMSG	IOERMSW	IOERNUM	IOERORA	ICERPEND	IOERREK	IOERREAD	IOERREW	IOERSIZE	IOERSTRT
	IOERSUPP	IOERVLD	IOERWRK	LOCK	NOP	PRGC	PROCIC	PRTC	PSA	RDEVBLOK	RDEVCUB	RDEVFTR	RDEVIOER
	RDEVNRDY	RDEVSTAT	RDEVTYPE	R0	R1	R10	R11	R12	R13	R14	R15	R2	R3
	R4	R5	R6	R7	R8	R9	SAVEAREA	SAVEREGS	SAVEWRK1	SAVEWRK4	SILI	SKIP	TIC
	TYP2401	TYP2415	TYP2420	TYP3410	TYP3420	UC	XRIGHT16	ZEROES					
DMKTCS	ADSPCH	AFREE	AFRET	APSTAT1	APTRAN	APTRLK	APUCPER	ASYSVM	BLANKS	BRING	CC	C1	DEFER
	DMKCVTBD	DMKCVTBH	DMKDSPCH	DMKERMSG	DMKFREE	DMKFRET	DMKIOSQR	DMKLOKSW	DMKPGTVG	DMKPTRAN	DMKQNTBL	DMKRPAGT	
	DMKSCNAU	DMKSCNRD	DMKSCNVS	FCBIO	FTR4WCGM	F1	F256	F3	F4095	F4096	IDA	IOBCAW	IOBFATAL
	IOBFLAG	IOBIRA	IOBLINK	IOBLOK	IOBMISC	IOBMISC2	IOBRSTRT	IOBSIZE	IOBSTAT	LOCK	NPRNAME	NPRPNT	NPRSTART
	NPRTBL	NPRVOL	PSA	RDEVBLOK	RDEVCODE	RDEVCURP	RDEVEXTN	RDEVFLAG	RDEVFSEP	RDEVFTR	RDEVIMAG	RDEVOWN	RDEVPURG
	RDEVSTA2	RDEVTYPE	RDEVXSEP	RSPDPAGE	RSPLCTL	RSPRPAGE	RSPRPAG2	RSPVPAGE	RSPXELCK	RSPXCHR	RSPXCMOD	RSPXCRWC	RSPXFCB

MODULE	EXTERNAL REFERENCES (LABELS AND MODULES)													
	R0	R1	R10	R11	R12	R13	R14	R15	R2	R3	R4	R5	R6	
	R7	R8	R9	SAVEAREA	SAVEREGS	SAVER11	SAVEWRK1	SAVEWRK2	SAVEWRK3	SAVEWRK4	SAVEWRK5	SAVEWRK6	SAVEWRK7	
	SAVEWRK8	SAVEWRK9	SFBCOPY	SFBFILID	SFBLOK	SFBSTART	SFBSTCPY	SFBUSER	SILI	SKIP	SPCHAR	SPCMOD	SPCOPYFG	
	SPFCB	SPFLAG1	SPFLSHC	SPLINK	SYSTEM	TIMEDISP	TYP2314	TYP3330	TYP3350	VMBLOK	VMSEG	XPAGNUM	ZEROES	
DMKTDK	ADSPCH	AFREE	AFRET	ALOCBLOK	ALOCCYL1	ALOCCYL2	ALOCMAP	ALOCPNT	CC	DMKDSPCH	DMKFREE	DMKFRET	DMKIOSQR	
	DMKPGTP0	DMKPGTP4	DMKPGTP5	DMKPGTT0	DMKPGTT4	DMKPGTT5	DMKPGT4P	DMKPGT4T	DMKPGT5P	DMKPGT5T	FTR70MB	F255	F256	
	IOBCP	IOBCYL	IOBFLAG	IOBLOK	IOBMISC	LOCK	PSA	RDEVALLN	RDEVELOK	RDEVFTR	RDEVLNKS	RDEVPNT	RDEVTYPE	
	R0	R1	R10	R11	R12	R13	R14	R15	R2	R3	R4	R5	R6	
	R7	R8	R9	SAVEAREA	SAVEREGS	SAVER0	SAVER1	SAVER8	SAVEWRK2	SEEK	SILI	TIC	TYP2314	
	TYP3330	TYP3340	TYP3350											
DMKTHI	AFREE	AFRET	APSTAT1	APUOPER	AQCNWT	ASYSVM	BLANKS	DFRET	DMKCVTBD	DMKCVTBH	DMKERMSG	DMKFREE	DMKFRET	
	DMKQCNWT	DMKSCHCA	DMKSCHCO	DMKSCHCU	DMKSCHEL	DMKSCHLI	DMKSCHRL	DMKSCHSC	DMKSCHS1	DMKSCHS2	DMKSCNFD	DMKSCNRD	DMKSCNVU	
	DMKTMRPT	F1	F3	F4	F60	F8	LOCK	NCRET	PREFIXE	PROCIO	PSA	RUNUSER	R0	
	R1	R10	R11	R12	R13	R14	R15	R2	R3	R4	R5	R6	R7	
	R8	R9	SAVEAREA	SAVEREGS	SAVER11	SAVEWRK1	SAVEWRK2	SAVEWRK3	SAVEWRK4	SAVEWRK6	VDEVELOK	VDEVREAL	VMACTDEV	
	VMBLOK	VMCLASSA	VMCLASSB	VMCLASSC	VMCLASSD	VMCLASSE	VMCLASSF	VMCLASSG	VMCLEVEL	VMCRDS	VMDEFSTK	VMDSTAT	VMELIG	
	VMEXWAIT	VMINQ	VMIOCNT	VMIOWAIT	VMLINS	VMPAGES	VMPDISK	VMPDRUM	VMPEND	VMPGREAD	VMPGWAIT	VMPGWRIT	VMPNCH	
	VMPNT	VMPSWAIT	VMQFPNT	VMQLEVEL	VMQ1	VMRSTAT	VMRUN	VMSTKO	VMTTIME	VMUSER	VMWSPROJ			
DMKTMR	ADSPCH	AFREE	AFRET	APSTAT1	APTRAN	APUOPER	ATMRSN	BALR14	BALR2	BRING	CHGREGS	CPCREG0	CPEXADD	
	CPEXBLOK	CPEXREGS	CPRUN	CPSTATUS	C0	C1	DEFER	DMKCVTAB	DMKDSPA	DMKDSPCH	DMKDSPRU	DMKFREE	DMKFRET	
	DMKLOKDF	DMKLOKSY	DMKPRGSM	DMKPSAFP	DMKPSASP	DMKPTRAN	DMKSCHN1	DMKSCHN2	DMKSCHRT	DMKSCHST	DMKSTKDE	DMKSTKIO	DMKVATEX	
	DMKVATRN	ECBLOK	EXTCCTRQ	EXTCPTMR	EXTCPTRQ	EXTCR9	EXTMASK	EXTPERAD	EXTSHCR0	F1	F4	F4095	F5	
	F60	F7	F8	LOCK	LOCKSAV	LPUADDR	PERSALT	PROBSTRT	PROETIME	PSA	R0	R1	R10	
	R11	R12	R13	R14	R15	R2	R3	R4	R5	R6	R7	R8	R9	
	SAVE	TRANMODE	TREXCR9	TREXPERA	TREXT	TRQBFPNT	TRQBLOK	TRQBQUE	TRQETOD	TRQBVAL	TYPE	VMAPTIME	VMBLOK	
	VMCPTIME	VMCPUTMR	VMDFTPNT	VMDSP	VMDSTAT	VMECEXT	VMESTAT	VMEXTCM	VMEXWAIT	VMGPRS	VMINQ	VMINST	VMINVPAG	
	VMPEND	VMPERCM	VMPERPND	VMPRGIL	VMPSTAT	VMPSW	VMPXINT	VMQLEVEL	VMQ1	VMRSTAT	VMSEG	VMTLEVEL	VMTMOUTQ	
	VMTMRINT	VMTRCTL	VMTREXT	VMTRPER	VMTTIME	VMVTIME	VMV370R	XINTBLOK	XINTNEXT	XINTPARM	XINTSIZE	XINTSORT	ZEROES	
DMKTRA	AFREE	AFRET	AQCNWT	COUNT	CSW	C1	DMKERMSG	DMKFREE	DMKFRET	DMKLOCKD	DMKLOCKQ	DMKQCNWT	DMKSCNFD	
	DMKTRCIT	DMKTRCPB	FFS	FLAG1	FLAG2	F3	F8	LOCK	MICELOK	MICVTMR	NORET	PSA	RUN	
	R0	R1	R11	R12	R13	R14	R15	R2	R3	R4	R5	R9	SAVEAREA	
	SAVEREGS	SAVER2	SAVEWRK1	SAVEWRK2	SAVEWRK7	TIMER	TREXANSI	TREXBRAN	TREXCCW	TREXCSW	TREXCTL	TREXINST	TREXIN1	
	TREXPRNT	TREXRUNF	TREXSIZE	TREXT	TREXTERM	VMBLOK	VMCFWAIT	VMEXWAIT	VMFSTAT	VMFVTMR	VMMADDR	VMMCR6	VMMVTMR	
	VMPSW	VMRON	VMRSTAT	VMSEG	VMTIMER	VMTLEVEL	VMTON	VMTRBRIN	VMTRCTL	VMTREX	VMTREXT	VMTRINT	VMTRIO	
	VMTRPER	VMTRPRG	VMTRPRV	VMTRSIO	VMTRSVC	WAIT	XRIGHT16							
DMKTRC	AFRET	APTRAN	APTRLK	AQCNWT	BLANKS	BRING	CPCREG0	C0	C1	DEFER	DMKATSCF	DMKCPMBK	DMKCVTBH	
	DMKFRET	DMKLOCKD	DMKLOCKQ	DMKNEMOP	DMKPSARR	DMKPSARS	DMKPSARX	DMKPSASC	DMKPSASP	DMKPTRAN	DMKPTRLK	DMKPTRUL	DMKQCNWT	
	DMKVATRN	DMKVSPRT	ECBLOK	EXTCR0	EXTMASK	EXTMODE	EXTSHCR0	FFS	F15	F2	F60	F8	INTSVCL	
	IOBCSW	IOBLOK	IOBRADD	IOBVADD	IOMASK	LOCK	NORET	PERMODE	PSA	R0	R1	R10	R11	
	R12	R13	R14	R15	R2	R3	R4	R5	R6	R7	R8	R9	SAVEAREA	
	SAVEREGS	SAVER0	SAVER1	SAVER2	SAVER4	SAVER5	SAVEWRK1	SAVEWRK2	SAVEWRK3	SAVEWRK4	SAVEWRK5	SAVEWRK6	SAVEWRK7	

MODULE	EXTERNAL REFERENCES (LABELS AND MODULES)												
	SAVEWRK8	SAVEWRK9	SVCNPSW	SVCOPSW	TRANMODE	TREXANSI	TREXBRAN	TREXBUFF	TREXCSW	TREXCTL1	TREXCTL2	TREXFLAG	TREXINST
	TREXIN1	TREXIN2	TREXLCNT	TREXNDSP	TREXNSI	TREXPRNT	TREXRUNF	TREXSIZE	TREXSVC1	TREXSVC2	TREXT	TREXTERM	TREXVAT
	VDEVBLOK	VDEVCSW	VMBLOK	VMCFWAIT	VMECEXT	VMESTAT	VMEXTCM	VMEXWAIT	VMGPRS	VMINVPAG	VMLOGOFF	VMPEND	VMPERPND
	VMPSTAT	VMPSW	VMRSTAT	VMSEG	VMTRBRIN	VMTRCTL	VMTREX	VMTREXT	VMTRINT	VMTRIO	VMTRPRG	VMTRPRV	VMTRSIO
	VMTRSVC	VMVCR0	VMV370R	WAIT	ZEROES								
DMKTRD	ADSPCH	AFREE	AFRET	APTRAN	AQCNWT	BLANKS	BRING	CAW	CLASDASD	CLASGRAF	CLASSPEC	CLASTERM	CLASURI
	CLASURO	CPCREG0	CPEXADD	CPEXBLOK	CPEXSIZE	CSW	C0	C1	DEFER	DMKCCWSB	DMKCFMBK	DMKCVTBH	DMKDSPCH
	DMKFREE	DMKFRET	DMKLOCKD	DMKLOCKQ	DMKNEMOP	DMKQCNWT	DMKSCNRD	DMKSCNRN	DMKSCNVN	DMKSCNVU	DMKSTKCP	DMKSYSRM	DMKVATRN
	DMKVSPRT	ECBLOK	EXTMODE	EXTSHCR0	FFS	F1	F15	F16	F2	F240	F3	F4	F60
	F8	IDA	IOBCAW	IOBCSW	IOBLOK	IOBRADD	IOBSTAT	LOCK	NORET	PSA	RCWCCW	RCWGEN	RCWPNT
	RCWRCNT	RCWTASK	RCWVCAW	RCWVCNT	R0	R1	R10	R11	R12	R13	R14	R15	R2
	R3	R4	R5	R6	R7	R8	R9	SAVEAREA	SAVEREGS	SAVER0	SAVER1	SAVER2	SAVEWRK1
	SAVEWRK2	SAVEWRK4	SAVEWRK6	SAVEWRK7	SAVEWRK8	SAVEWRK9	TRANMODE	TREXANSI	TREXBRAN	TREXBUFF	TREXCCW	TREXCTL1	TREXCTL2
	TREXFLAG	TREXINST	TREXIN1	TREXIN2	TREXLCNT	TREXNDSP	TREXPRNT	TREXRUNF	TREXSIZE	TREXSVC1	TREXSVC2	TREXT	TREXTERM
	TREXVAT	VDEVBLOK	VDEVDED	VDEVREAL	VDEVSTAT	VDEVTYPC	VMBLOK	VMCFWAIT	VMECEXT	VMESTAT	VMEXTCM	VMEXWAIT	VMINST
	VMINVPAG	VMLOGOFF	VMPSW	VMRSTAT	VMSEG	VMSTOR	VMTRBRIN	VMTRCTL	VMTREXT	VMTRINT	VMTRSIO	X2048BND	ZEROES
DMKTRK	ADSPCH	AFREE	AFRET	APSTAT1	APTRAN	BRING	CC	CD	C1	DEFER	DMKCCWSB	DMKDSPCH	DMKFREE
	DMKFRET	DMKIOSQR	DMKIOSQV	DMKPTRAN	DMKUNTRN	FFS	FINIS	F1	F3	F6	F8	IL	IOBALISK
	IOBCAW	IOBCC3	IOBCP	IOBCSW	IOBFATAL	IOBFLAG	IOBHVC	IOBIOER	IOBIRA	IOBLOK	IOBMISC2	IOBRCAW	IOBRCNT
	IOBRSTRT	IOBSIOF	IOBSIZE	IOBSPEC	IOBSTAT	IOBUSER	IOERADR	IOERALTR	IOERELOK	IOERCMD	IOERCSW	IOERDATA	IOERDW
	IOEREXT	IOERFLG2	IOERFLG3	IOERLOC	IOERPNT	IOERRDR0	IOERSIZE	IOERWRK	LOCK	PCI	PROCIO	PSA	RCWADDR
	RCWCCNT	RCWCCW	RCWCNT	RCWCOMND	RCWCTL	RCWFLAG	RCWGEN	RCWHEAD	RCWINVL	RCWPNT	RCWRCNT	RCWREL	RCWTASK
	RDEVBLOK	RDEVIOER	R0	R1	R10	R11	R12	R13	R14	R15	R2	R3	R4
	R5	R6	R7	R8	R9	SAVEAREA	SAVEREGS	SAVER1	SAVER7	SAVER8	SAVEWRK1	SAVEWRK2	SAVEWRK3
	SAVEWRK4	SAVEWRK5	SAVEWRK6	SAVEWRK7	SAVEWRK8	SILI	SKIP	UC	VDEVELCK	VDEVCSW	VDEVREAL	VMBLOK	VMSEG
	XRIGHT16	ZEROES											
DMKTRM	APSTAT1	F7	LOCK	PROCIO	PSA	RDEVATOF	RDEVBLOK	RDEVCORR	RDEVFLAG	RDEVIDNT	RDEVPTTC	RDEVTFLG	RDEVTMCD
	R0	R1	R11	R12	R13	R2	R3	R4	R5	R8	SAVEAREA	SAVEREGS	
DMKUCB	CC	LOCK	NUM	SILI									
DMKUCC	CC	LOCK	NUM	SILI									
DMKUCS	CC	LOCK	NUM	SILI									
DMKUDR	ACORETBL	ADSPCH	AFREE	AFRET	ALARM	APTRAN	AQCNWT	ARIODV	ASYSLC	ASYSVM	BLANKS	BRING	CC
	CCW1	CORFPNT	CORPGPNT	CORTABLE	C1	DEFER	DMKDSPCH	DMKFREE	DMKFRET	DMKIOSQR	DMKLOCKD	DMKLOCKQ	DMKPGTVG
	DMKPGTVR	DMKPTRAN	DMKPTRFT	DMKQCNWT	DMKRPAGT	DMKSYSOC	DMKSYSOW	DMKSYSPL	DMKSYSUD	F1	F256	F4096	F8
	IOBCAW	IOBCP	IOBFATAL	IOBFLAG	IOBIRA	IOBLOK	IOBMISC	IOBMISC2	IOBSIZE	IOBSTAT	IOBUSER	IOERETN	NOADD
	NORET	OPERATOR	OWNDLIST	OWNDRDEV	OWNDVSER	PAGINVAL	PSA	R0	R1	R10	R11	R12	R13
	R14	R15	R2	R3	R4	R5	R6	R7	R8	R9	SAVEAREA	SAVEREGS	SAVER0
	SAVER2	SAVEWRK2	SILI	SYSLOCS	SYSTEM	UDBFBLOK	UDBFDASD	UDBFVADD	UDBFWORK	UDEVADD	UDEVBLOK	UDEVDASD	UDEVDISP
	UDEVSIZE	UDIRBLOK	UDIRDASD	UDIRDISP	UDIRSIZE	UDIRUSER	UMACBLCK	UMACDASD	UMACDISP	UMACDVCT	UMACSIZE	VMBLOK	VMESTAT

MODULE	EXTERNAL REFERENCES (LABELS AND MODULES)												
	VMEXTCM	VMPAGES	VMPSW	VMSEG	ZEROES								
DMKUDU	ACORETBL	AFREE	AFRET	APTRAN	ASYSLC	ASYSOP	ASYSVM	BLANKS	BRING	BUFCNT	BUFFER	BUFNXT	BUFSIZE
	CORSWPNT	CORTABLE	C1	DEFER	DMKCPVAC	DMKFREE	DMKFRET	DMKLOCKD	DMKLOCKQ	DMKPGTVG	DMKPGTVR	DMKPTRAN	DMKRPAGT
	DMKRPAPT	DMKSCNAU	DMKSYSPL	DMKSYSUD	F2	F240	IOERETN	PSA	R0	R1	R10	R11	R12
	R13	R14	R15	R2	R3	R4	R5	R6	R7	R8	R9	SAVEAREA	SAVEREGS
	SWPCYL	SWPFLAG	SWPRECMP	SYSLOCS	SYSTEM	UCNTRL	UCNTRLSZ	UCURPASS	UDASIDEV	UDASDDIR	UDASDMAC	UDEVAD	UDEVADD
	UDEVBLOK	UDEVCODE	UDEVDASD	UDEVDISP	UDEVF	UDEVLM	UDEVLCNG	UDEVLR	UDEVLW	UDEVMODE	UDEVMR	UDEVMW	UDEVPASR
	UDEVRR	UDEVSTAT	UDEVVRR	UDEVWR	UDIRAD	UDIRBLOK	UDIRDASD	UDIRDISP	UDIRF	UDIRPASS	UDIRSIZE	UDISPDEV	UDISPMAC
	UFLAGS	UIPARMS	UIPARMSZ	ULOCDVAD	UMACACCT	UMACAD	UMACAFF	UMACBLOK	UMACCLEV	UMACCORE	UMACDIST	UMACF	UMACFFON
	UMACIPL	UMACLEND	UMACMCOR	UMACOPT	UMACPRIR	UMACPUID	UMDISKAD	UMDISKMD	UMDISKMP	UMDISKRP	UMDISKWP	UNEWPASS	UNOUPF
	UOBJVMBK	UOP	UOPTIONS	UPRIOR	UPRIV	URECMP	URETCODE	URPAGDEV	URPAGDIR	URPAGMAC	USTORAGE	USVDASD	UTESTMD
	UUSERID	UVMBLOK	UVPAGBUF	UVPAGDIR	UWORK	VMACNT	VMBLOK	VMSEG	ZEROES				
DMKUNT	ACORETBL	ADSPCH	AFREE	AFRET	BALRSAVE	CCC	CD	CDC	CHC	CORPGPNT	CORTABLE	CPEXBLOK	CPEXFPNT
	DATACHK	DE	DMKDSPCH	DMKFREE	DMKFREST	DMKFRET	DMKFRETL	DMKPTRFT	DMKPTRPL	DMKPTRUL	DMKSCNVD	DMKSTKCP	DMKSTKIO
	DMKSYSRM	DMKTRKFP	DMKVIOIN	FFS	FTR35MB	F0	F1	F15	F16	F240	F4	F7	F9
	IDA	IFCC	IOBALTSK	IOBCAW	IOBCC3	IOBCLN	IOBCSW	IOBFLAG	IOBIRA	IOBLINK	IOBLOK	IOBMISC	IOBRES
	IOBSIZE	IOBSPEC2	IOBSTAT	IOBUNREL	IOBUSER	IOBVADD	IOERBLCK	ICERCYLR	IOERDATA	IOERFLG2	IOERLEN	LOCK	PRGC
	PRTC	PSA	RCWADDR	RCWCCNT	RCWCCW	RCWCNT	RCWCCMND	RCWCTL	RCWFLAG	RCWGEN	RCWHMR	RCWIO	RCWPNT
	RCWRCNT	RCWSHR	RCWTASK	RCWVCAW	RCW2311	RDEVBLOK	RDEVFTR	RDEVMDL	R0	R1	R10	R11	R12
	R13	R14	R15	R2	R3	R4	R5	R6	R7	R8	R9	SAVEAREA	SAVEREGS
	SAVER11	SAVER8	SAVEWRK1	SAVEWRK2	SAVEWRK3	SAVEWRK5	SAVEWRK6	SAVEWRK9	SKIP	TYP2305	TYP3330	TYP3340	TYP3350
	UC	VDEVBLOK	VDEVCSW	VDEVFLAG	VDEVFLG2	VDEVLINK	VDEVODE	VDEVREAL	VDEVRELN	VDEVRES	VDEVRRB	VDEVTYPE	VDEVUSER
	VDEV2318	VMBLOK	VRRBLOK	VRRCPEX	VRRRES	VRRSTAT	XPAGNUM	X2048BND	ZEROES				
DMKUSO	ACCTLENG	ADSPCH	AFREE	AFRET	APTRLK	AQCNWT	ARSPPR	ARSPPU	ARSPRD	ASYSLC	ASYSOP	ASYSVM	BLANKS
	CLASGRAF	CLASSPEC	CLASTERM	CPEXADD	CPEXBLOK	CPEXR0	CPEXR11	CPEXR12	CPEXR2	CPEXSIZE	DELPAGES	DELSEGS	DMKACOFF
	DMKACOTM	DMKBLDRL	DMKCFPRR	DMKCVTBD	DMKCVTBH	DMKCVTDT	DMKDSPCH	DMKERMSG	DMKFREE	DMKFREHI	DMKFRELO	DMKFRENP	DMKFRERS
	DMKFRET	DMKLOCKD	DMKLOCKQ	DMKPAGSK	DMKPERT	DMKPGSPO	DMKPGSPP	DMKPGTP5	DMKPTRLK	DMKPTRPW	DMKPTRRL	DMKPTRRU	DMKPTRUL
	DMKQCNWT	DMKSCHAU	DMKSCHDL	DMKSCHRT	DMKSCNAU	DMKSCNFD	DMKSCNRD	DMKSCNRN	DMKSCNVD	DMKSSSUS	DMKSSSVM	DMKSTKCP	DMKSYSDW
	DMKSYSNM	DMKSYSTI	DMKTRCND	DMKVATBC	DMKVDREL	DMKVMASH	DMKVSEWA	ECBLOK	EXTCCTRQ	EXTCPTRQ	EXTSIZE	FFS	F15
	F8	HOLD	IOBLOK	IOBMISC	LASTUSER	LOCK	LOGDRCP	LOGHOLD	MICSIZE	MSSPRES	NORET	OPERATOR	PAGTONLY
	PRIORITY	PSA	PSAMSS	RDEVBLOK	RDEVPAGE	RDEVPNT	RDEVRECS	RDEVTYPC	RDEVTYPE	RECSIZE	RUNUSER	R0	R1
	R10	R11	R12	R13	R14	R15	R2	R3	R4	R5	R6	R7	R8
	R9	SAVEAREA	SAVEREGS	SAVERETN	SAVEWRK1	SAVEWRK2	SAVEWRK3	SAVEWRK4	SFBFLAG	SFBINUSE	SFBLOK	SFBSIZE	STACKVM
	START	SYSLOCS	TIMEDISP	TREXSIZE	TRQBFPNT	TRQBLOK	TRQBSIZE	TYPBSC	TYP3705	VCHBLOK	VCHSIZE	VCUBLOK	VCUDVTBL
	VCUSIZE	VDEVADD	VDEVBLOK	VDEVSIZE	VMACCOUN	VMACNT	VMACOUNT	VMAEXP	VMBLOK	VMBSIZE	VMCHCNT	VMCHSTRT	VMCHTBL
	VMCOMND	VMCUCNT	VMDELAY	VMDFTPNT	VMDISC	VMDVCNT	VMDVSTRT	VMECEXT	VMINHMIG	VMKILL	VMLOGOFF	VMLOGON	VMMICRO
	VMMLEVEL	VMMSGON	VMNDCNT	VMOSTAT	VMPFUNC	VMPGWAIT	VMPNT	VMPSTAT	VMPSW	VMQLEVEL	VMQSTAT	VMRPAGE	VMRSTAT
	VMSEG	VMSHR	VMSIZE	VMSWPMIG	VMSYSOP	VMTERM	VMTRCTL	VMTREXT	VMTRMID	VMTRPER	VMTRQBLK	VMUSER	VMVTERM
	VMV370R	VMWNGON	WAIT	ZEROES									
DMKVAT	ADSPCH	AFREE	AFRET	APSTAT1	APSTAT2	APTRAN	APUCPER	ASYSVM	BALRSAVE	BALR12	BALR13	BALR14	BRING
	CPCREG0	CPEXADD	CPEXBLOK	CPEXREGS	CPPTLBR	C1	DEFER	DMKATSCF	DMKDSPCH	DMKDSPRU	DMKFREE	DMKFRET	DMKLOKDF
	DMKLOKSY	DMKPERIL	DMKPRGSM	DMKPTRAN	DMKPTRPW	DMKSTKDE	ECBLOK	EXTARCH	EXTCOPY	EXTCR0	EXTCR1	EXTMODE	EXTSEGLN

MODULE	EXTERNAL REFERENCES (LABELS AND MODULES)												
	EXTSHCR0	EXTSHCR1	EXTSHLEN	EXTSHSEG	EXTSTOLD	EXTVSEGS	F1	F16	F4	F8	INTPR	LOCK	LPUADDR
	PAGSWP	PAGTABLE	PGADDR	PGBLOK	PGBSIZE	PGPNT	PREFIXB	PSA	R0	R1	R10	R11	R12
	R13	R14	R15	R2	R3	R4	R5	R6	R7	R8	R9	SAVEAREA	SAVEREGS
	SAVERETN	SAVER0	SAVER1	SAVER12	SAVER13	SAVER2	SAVER3	SWPFLAG	SWPSHR	SWPTABLE	TRANMODE	TREXADD	TYPE
	VMBADCR0	VMBLOK	VMDFTPNT	VMECEXT	VMESTAT	VMEXWAIT	VMINVFAG	VMINVSEG	VMIOPND	VMNEWCR0	VMOSTAT	VMPAGEX	VMPEND
	VMPERPND	VMPGPND	VMPGPNT	VMPGWAIT	VMPRGIL	VMPSTAT	VMPSW	VMRSTAT	VMSEG	VMSHADT	VMSHR	VMTRCTL	VMTRPER
	XRIGHT16												
DMKVCA	ADSPCH	AFREE	AFRET	APSTAT1	APUOPER	AQCNWT	ATTN	BALRSAVE	BALR14	BALR15	BALR2	BALR3	BALR9
	BLANKS	BUSY	CC	CD	CE	CHBATTN	CHBCENT	CHBCNTL	CHBEOFL	CHBHIO	CHBMNOP	CHBM370	CHBRDBK
	CHBREAD	CHBREST	CHBSIZE	CHBWAIT	CHBWEOF	CHBWRIT	CHXBLCK	CHXCMDB	CHXCMDT	CHXCNCT	CHXDATN	CHXFLAG	CHXIDAW
	CHXNCCW	CHXOTHR	CHXPKEY	CHXRCNT	CHXSTAT	CHXYADD	CHYBLCK	CHYCMDB	CHYCMDT	CHYCNCT	CHYDATN	CHYFLAG	CHYIDAW
	CHYNCCW	CHYRCNT	CHYSTAT	CHYXADD	CPEX	CPEXADD	CPEXBLCK	CPEXFPNT	CPEXR0	CPEXR12	CPEXSIZE	CPWAIT	DATAEND
	DE	DMKCVTBH	DMKDIBSM	DMKDSPCH	DMKFREE	DMKFRET	DMKLOKSW	DMKQCNWT	DMKSCHDL	DMKSCNVU	DMKSTKCP	DMKSTKIO	DMKSYSRM
	DMKTRDSI	DMKVIOIN	FFS	FREESAVE	F1	F2	F240	IDA	IL	INTREQ	IOBCAW	IOBCC1	IOBCC3
	IOBCSW	IOBFLAG	IOBIOER	IOBIRA	IOBLINK	IOBLOK	IOBRES	IOBRSTRT	IOBSIZE	IOBSTAT	IOBUSER	IOBVADD	IOERBLOK
	IOERCCW	IOERCSW	IOERDATA	IOERLEN	IOERSIZE	LOCK	NORET	PCI	PCIF	PRGC	PRTC	PSA	RCWADDR
	RCWCCW	RCWCNT	RCWCOMND	RCWCTL	RCWFLAG	RCWINVL	RUNUSER	R0	R1	R10	R11	R12	R13
	R14	R15	R2	R3	R4	R5	R6	R7	R8	R9	SAVEAREA	SAVEREGS	SAVERETN
	SAVER0	SAVER1	SAVER10	SAVER11	SAVER12	SAVER5	SAVEWRK1	SAVEWRK2	SAVEWRK3	SAVEWRK4	SAVEWRK6	SAVEWRK9	SILI
	SKIP	TEMPR0	TEMPR2	TEMPR3	TEMPR4	TEMPR5	TIMEDISP	TYPCTCA	UC	UE	VDEVBLOK	VDEVCCW1	VDEVINTS
	VDEVIOCT	VDEVNRDY	VDEVREAL	VDEVSTAT	VMBLOK	VMDVSTRT	VMIOWAIT	VMLOGOFF	VMRSTAT	VMRUN	VMTRCTL	VMTRSIO	VMUSER
	ZEROES												
DMKVCH	AFREE	AFRET	APSTAT1	APTRAN	APUOPER	AQCNWT	ARIOCU	ARIODV	ASYSCP	ASYSVM	BRING	CLASDASD	CLASGRAF
	CLASSPEC	CLASTAPE	CLASTERM	CLASURI	CLASURO	C1	DEFER	DMKCVTEH	DMKERMSG	DMKFREE	DMKFRERC	DMKFRET	DMKLOCKD
	DMKLOCKQ	DMKLOKSW	DMKPTRAN	DMKQCNWT	DMKSCNAU	DMKSCNRA	DMKSCNRU	DMKSCNVU	DMKVDREL	DMKVDSAT	FFS	F1	LOCK
	NORET	OPERATOR	PROCIO	PSA	RCHBLOK	RCHCUTBL	RCHDED	RCHSTAT	RCUBLOK	RCUCHA	RCUCHAOF	RCUCHBOF	RCUCHCOF
	RCUCHD	RCUCHDOF	RCUDISA	RCUDVTBL	RCUSTAT	RDEVADD	RDEVATT	RDEVBLOK	RDEVBUSY	RDEVDED	RDEVDISA	RDEVDRAN	RDEVENAB
	RDEVFLAG	RDEVOWN	RDEVRCVY	RDEVRSVD	RDEVSCED	RDEVSPL	RDEVSTAT	RDEVSYS	RDEVTYPC	RDEVTYPE	R0	R1	R10
	R11	R12	R13	R14	R15	R2	R3	R4	R5	R6	R7	R8	R9
	SAVEAREA	SAVEREGS	SAVER10	SAVER11	SAVER2	SAVEWRK1	SAVEWRK2	SAVEWRK4	SAVEWRK5	SAVEWRK6	SAVEWRK8	SAVEWRK9	SYSTEM
	TIMEDISP	TYP3705	VCHADD	VCHBLOK	VCHBMX	VCHCUTBL	VCHDED	VCHSEL	VCHSIZE	VCHSTAT	VCHTYPE	VCUADD	VCUBLOK
	VCUDVTBL	VCUSIZE	VDEVADD	VDEVBLOK	VDEVFLAG	VDEVLINK	VDEVSIZE	VDEVTDSK	VDEVTYPC	VMBLOK	VMCHCNT	VMCHSTRT	VMCHTBL
	VMCUCNT	VMCUSTRT	VMDVCNT	VMDVSTRT	VMFBMX	VMFSTAT	VMSEG	VMUSER	ZEROES				
DMKVCN	ADSPCH	AFREE	AFRET	ALARM	APTRAN	AQCNWT	BLANKS	BRING	BUFCNT	BUFIN	BUFNXT	CC	CD
	CE	CLASGRAF	CLASTERM	CMDREJ	CSW	C1	DE	DEFER	DMKCFMAT	DMKCFMBK	DMKCFMEN	DMKDSPCH	DMKFREE
	DMKFRET	DMKGRTDS	DMKGRTIN	DMKPSACC	DMKPSASC	DMKPTRAN	DMKQCNRD	DMKQCNWT	DMKSCNVU	DMKTBLUP	DMKVIOMK	DMKVMASH	EDIT
	F1	F256	F3	F4	F7	F8	IDA	IL	INHIBIT	INTREQ	LOCK	NOAUTO	NORET
	NOTIME	PCI	PCIF	PRGC	PRIORITY	PRTC	PSA	RDEVBLOK	RDEVGRTY	RDEVTYPC	RDEVTYPE	R0	R1
	R10	R11	R12	R13	R14	R15	R2	R3	R4	R5	R6	R7	R8
	R9	SILI	SKIP	TIC	TYPBSC	TYP3210	UC	UE	VCHADD	VCHBLOK	VCHCUINT	VCONADDR	VCONBFSZ
	VCONBUF	VCONCAW	VCONCCW	VCONCNT	VCONCOMD	VCONCTL	VCONDWC	VCONFLAG	VCONIDAP	VCONRBSZ	VCONRBUF	VCONRCNT	VCONWBSZ
	VCONWBUF	VCONWCNT	VCUADD	VCUBLOK	VCUCEPND	VCUDVINT	VCUSHRD	VCUSTAT	VCUTYPE	VDEVADD	VDEVATTN	VDEVBLOK	VDEVBUSY
	VDEVCCW1	VDEVCFLG	VDEVCHAN	VDEVCHBS	VDEVCON	VDEVCSPL	VDEVCSW	VDEVFLAG	VDEVINTS	VDEVIOCT	VDEVKEY	VDEVNRDY	VDEVPEND

MODULE	EXTERNAL REFERENCES (LABELS AND MODULES)												
	VDEVSFLG	VDEVSNSE	VDEVSTAT	VDEVTERM	VDEVTIC	VDEVTRAN	VDEVTYPC	VDEVTYPE	VDEVVCF	VMBLOK	VMCF	VMCHSTRT	VMCUSTRT
	VMDISC	VMDSTAT	VMDVSTRT	VMESTAT	VMEXTCM	VMEXWAIT	VMGENIO	VMIDLE	VMICINT	VMIOPND	VMLOGOFF	VMMLEVEL	VMMLINED
	VMMSTMF	VMOSTAT	VMPEND	VMPRIDSP	VMPSW	VMQSTAT	VMRBSC	VMRSTAT	VMSEG	VMTERM	VMTIO	XRIGHT16	
DMKVDA	AFREE	AFRET	APSTAT1	APUOPER	AQCNWT	ASYSVM	BLANKS	CLASDASD	CLASSPEC	CLASTERM	CLASURI	CLASURO	CPEXBLOK
	CPEXR1	DEVADDR	DMKCVTBD	DMKCVTBH	DMKERMSG	DMKFREE	DMKFRET	DMKLOCKD	DMKLOCKQ	DMKLOKSW	DMKQCNWT	DMKSCNAU	DMKSCNRD
	DMKSCNRN	DMKSCNRU	DMKSCNVS	DMKSCNVU	DMKSSSAS	DMKSSSMQ	DMKSSSVA	DMKSTKCP	DMKSYSOC	DMKSYSOW	DMKVCHDC	DMKVDCAL	DMKVDCPS
	DMKVDCSC	DMKVDEDC	DMKVDSAT	ERROR	FFS	F8	LASTLINE	LOCK	MSSFLAGS	MSSNEXT	MSSSIZE	MSSTASK1	MSSTASK3
	NORET	NOTRESP	OPERATOR	OWNDLIST	OWNDPREF	OWNDVSER	PROCIC	PSA	RANGE	RDEVADD	RDEVBLOK	RDEVBUSY	RDEVDED
	RDEVDISA	RDEVDRAN	RDEVENAB	RDEVFLAG	RDEVFTR	RDEVLNKS	RDEVMCUT	RDEVOWN	RDEVPREF	RDEVPREP	RDEVSER	RDEVSPL	RDEVSTAT
	RDEVSYS	RDEVTYPC	RDEVTYPE	RDEVUSER	RDEV333V	R0	R1	R10	R11	R12	R13	R14	R15
	R2	R3	R4	R5	R6	R7	R8	R9	SAVEAREA	SAVEREGS	SAVER11	SAVER2	SAVESIZE
	SAVEWRK1	SAVEWRK2	SAVEWRK3	SAVEWRK5	SAVEWRK6	SAVEWRK7	SAVEWRK8	SAVEWRK9	SYSVIRT	TIMEDISP	TYPCTCA	TYP2305	TYP3330
	UDEVADD	UDEVBLOK	UDEVMODE	UDEVW	VCHADD	VCHBLOK	VCHDED	VCHSTAT	VCUBLOK	VCUDVTBL	VDEVBLOK	VDEVCATT	VDEVFLG2
	VDEVSTAT	VIRTUAL	VMBLOK	VMOSTAT	VMSYSOP	VMUSER	ZEROES						
DMKVDC	ADSPCH	AFREE	AFRET	ALOCBLOK	ALOCCYL1	ALOCCYL2	ALOCMAP	ALOCMAX	ALOCNTMP	ALOCPNT	ALOCUSED	ARIODV	ASYSVM
	BALR1	BALR14	BALR6	BLANKS	BUFFER	BUFNXT	CLASDASD	CLASGRAF	CLASSPEC	CLASTAPE	CLASTERM	CLASURI	CLASURO
	CPEXBLOK	CPEXR0	CPEXSIZE	DEVADDR	DMKCFCSC	DMKCVTHB	DMKDSPCH	DMKFREE	DMKFRERC	DMKFRET	DMKIOSQR	DMKPGTP0	DMKPGTP4
	DMKPGTP5	DMKPGTTM	DMKPGTT0	DMKPGTT4	DMKPGTT5	DMKPGT4P	DMKPGT4T	DMKPGT5P	DMKPGT5T	DMKPGT90	DMKSCNAU	DMKSCNFD	DMKSCNRU
	DMKSCNVU	DMKSYSOW	DMKVIOMK	FFS	FTRVIRT	FTR35MB	F0	F1	F10	F3	F4	F6	F7
	F8	F9	IOBCC3	IOBCP	IOBCSW	IOBFLAG	IOBIOER	IOBIRA	IOBLOK	IOBMISC	IOBMISC2	IOBSIZE	IOBSPEC
	IOBSTAT	IOBTIO	IOBUSER	IOERBLOK	IOERDATA	IOEREXT	IOERSIZE	LOCK	OWNDLIST	OWNDRDEV	PSA	RANGE	RDEVADD
	RDEVALLN	RDEVBLOK	RDEVCODE	RDEVDED	RDEVDISA	RDEVDRAN	RDEVENAB	RDEVFLAG	RDEVFTR	RDEVMDL	RDEVOWN	RDEVPNT	RDEVPREF
	RDEVRCVY	RDEVRSVD	RDEVSPL	RDEVSTAT	RDEVSYS	RDEVTYPC	RDEVTYPE	R0	R1	R10	R11	R12	R13
	R14	R15	R2	R3	R4	R5	R6	R7	R8	R9	SAVEAREA	SAVEREGS	SAVER0
	SAVER10	SAVER13	SAVER2	SAVEWRK1	SAVEWRK2	SAVEWRK3	SAVEWRK5	SAVEWRK6	SAVEWRK7	SAVEWRK8	SAVEWRK9	TAPE	TYPCTCA
	TYP2305	TYP2314	TYP3330	TYP3340	TYP3350	TYP3705	UC	VCHBLOK	VCHCUTBL	VCHDED	VCHSIZE	VCHSTAT	VCUBLOK
	VCUDVTBL	VCUSIZE	VDEVADD	VDEVBLOK	VDEVFLAG	VDEVLINK	VDEVSIZE	VDEVTDSK	VDEVTYPC	VMBLOK	VMCHCNT	VMCHSTRT	VMCLASSB
	VMCLEVEL	VMCUCNT	VMCUSTRT	VMDVCNT	VMDVSTRT	VMPSTAT	VMUSER	VMV370R	ZEROES				
DMKVDD	AFREE	AFRET	APSTAT1	APUOPER	AQCNWT	ASYSVM	BLANKS	CLASDASD	CLASTAPE	CLASURI	CLASURO	DE	DEVADDR
	DFRET	DMKCVTBD	DMKCVTBH	DMKDSBRD	DMKERMSG	DMKFREE	DMKFRET	DMKLOCKD	DMKLOCKQ	DMKLOKSW	DMKQCNWT	DMKSCNAU	DMKSCNRD
	DMKSCNRN	DMKSCNRU	DMKSCNVN	DMKSCNVU	DMKSTKIO	DMKVCHDC	DMKVDCSC	DMKVDREL	ERROR	FFS	F8	IOBCSW	IOBIRA
	IOBLINK	IOBLOK	IOBRADD	IOBSIZE	IOBSPEC	IOBUNSL	IOBUSER	LOCK	NORET	OPERATOR	PROCIO	PSA	RANGE
	RDEVADD	RDEVATT	RDEVBLOK	RDEVDED	RDEVDISA	RDEVFLAG	RDEVLNKS	RDEVMOUT	RDEVCWN	RDEVSER	RDEVSTAT	RDEVSYS	RDEVTYPC
	RDEVTYPE	RDEVUSER	R0	R1	R10	R11	R12	R13	R14	R15	R2	R3	R4
	R5	R6	R7	R8	R9	SAVEAREA	SAVEREGS	SAVER11	SAVER2	SAVEWRK1	SAVEWRK2	SAVEWRK3	SAVEWRK4
	SAVEWRK5	SAVEWRK6	SAVEWRK7	SAVEWRK8	SAVEWRK9	TIMEDISP	TYP2305	VCHADD	VCHBLOK	VCHCUTBL	VCHDED	VCHSTAT	VCUADD
	VCUBLOK	VCUDVTBL	VDEVADD	VDEVBLOK	VDEVCATT	VDEVDED	VDEVREAL	VDEVSFLG	VDEVSTAT	VDEVSVC	VDEVTYPC	VMBLOK	VMCHTBL
	VMKILL	VMLOGOFF	VMLOGON	VMMIMSG	VMMLVL2	VMOSTAT	VMRSTAT	VMSYSOP	VMUSER				
DMKVDE	ADSPCH	AFREE	AFRET	APSTAT1	BLANKS	CC	CL	CLASDASD	CLASTAPE	CLASTERM	CPEXBLOK	CPEXR0	CPEXR13
	CPEXR5	CPEXSIZE	DMKDSPCH	DMKFREE	DMKFRET	DMKIOSQR	DMKSCNRU	FTRRPS	FTRVIRT	FTR35MB	FTR70MB	F1	IOBCAW
	IOBCC1	IOBCC3	IOBCP	IOBCSW	IOBFATAL	IOBFLAG	IOBIOER	IOBIRA	IOBLOK	IOBMISC	IOBMISC2	IOBSIZE	IOBSPEC
	IOBSTAT	IOBTIO	IOBUSER	IOERBLOK	IOERDATA	IOEREXT	IOERSIZE	ICERSNSZ	LOCK	PROCIO	PSA	RDEVBLOK	RDEVDISA

MODULE	EXTERNAL REFERENCES (LABELS AND MODULES)												
	RDEVFTR	RDEVNRDY	RDEVSER	RDEVSTAT	RDEVTYPC	RDEVTYPE	R0	R1	R10	R11	R12	R13	R14
	R15	R2	R3	R4	R5	R6	R7	R8	R9	SAVEAREA	SAVEREGS	SAVER10	SAVER13
	SAVER2	SAVEWRK1	SILI	SKIP	TYPBSC	TYP2305	TYP3340	UC					
DMKVDR	AFREE	AFRET	APSTAT1	APTRAN	AQCNWT	ASYSVM	BLANKS	BRING	CLASDASD	CLASSPEC	CLASTAPE	CLASTERM	CLASURI
	CLASURO	C1	DEFER	DMKACODV	DMKCPPRD	DMKCVTBH	DMKFREE	DMKFRET	DMKIOSQR	DMKIOSRW	DMKPTRAN	DMKPTRPW	DMKQCNWT
	DMKSCNRD	DMKSCNRN	DMKTDKRL	DMKVCARS	DMKVSPCO	DMKVSPCR	FFS	F1	IOBCAW	IOBFLAG	IOBIRA	IOBLOK	IOBRELCU
	IOBSIZE	IOBUSER	LOCK	NORET	OPERATOR	PROCIO	PSA	RCWCCNT	RCWCCW	RCWHEAD	RCWTASK	RDEVADD	RDEVATT
	RDEVBLOK	RDEVCURP	RDEVDED	RDEVDELP	RDEVFLAG	RDEVLNKS	RDEVMOUT	RDEVOVLY	RDEVSTAT	RDEVSYS	RDEVTMAT	RDEVTYPC	RDEVTYPE
	RDEVUSER	R0	R1	R10	R11	R12	R13	R14	R15	R2	R3	R4	R6
	R8	SAVEAREA	SAVEREGS	SAVER8	SAVEWRK2	SAVEWRK3	SAVEWRK4	SAVEWRK6	SAVEWRK9	SILI	SYSTEM	TYPCTCA	TYP1052
	TYP2305	TYP3211	TYP3800	VCONBFSZ	VCONBUF	VCONCTL	VCONRBSZ	VCONREUF	VCONSIZE	VCONWESZ	VCONWBUF	VDEVBLOK	VDEVBND
	VDEVCATT	VDEVCON	VDEVDED	VDEVDET	VDEVEXTN	VDEVFCBK	VDEVFLAG	VDEVFLG2	VDEVLINK	VDEVREAL	VDEVRELN	VDEVRRB	VDEVRRF
	VDEVSPL	VDEVSTAT	VDEVTDSK	VDEVTYPC	VDEVTYPE	VFCBSIZE	VMBLOK	VMDVSTRT	VMSEG	VMUSER	VMVTERM	VRRSIZE	VSPXBLOK
	VSPXLEN	ZEROES											
DMKVDS	AFREE	AFRET	APSTAT1	BALR1	BLANKS	CLASDASD	CLASGRAF	CLASSPEC	CLASTAPE	CLASTERM	CLASURI	CLASURO	DMKCVTAB
	DMKCVTBH	DMKERMSG	DMKFREE	DMKFRERC	DMKFRET	DMKSCNLI	DMKSCNRD	DMKSCNRU	DMKSCNVU	DMKSPLDL	DMKSYSCK	DMKTDKGT	FFS
	FTRRSRL	F8	LOCK	NICSIZE	PROCIO	PSA	RDEVATT	RDEVBLOK	RDEVDED	RDEVDELP	RDEVDISA	RDEVDRAN	RDEVENAB
	RDEVEPLN	RDEVFLAG	RDEVFTR	RDEVLNCP	RDEVLNKS	RDEVMAX	RDEVMCUT	RDEVNICL	RDEVCWN	RDEVRCVY	RDEVRSVD	RDEVSPL	RDEVSTAT
	RDEVSYS	RDEVTMAT	RDEVTYPC	RDEVTYPE	RDEVUSER	R0	R1	R10	R11	R12	R13	R14	R15
	R2	R3	R4	R5	R6	R7	R8	R9	SAVEAREA	SAVEREGS	SAVER1	SAVER2	SAVER8
	SAVEWRK1	SAVEWRK2	SAVEWRK3	SAVEWRK4	SAVEWRK6	SAVEWRK7	SAVEWRK9	TYPBSC	TYPCTCA	TYP1052	TYP2305	TYP2311	TYP3210
	TYP3277	TYP3278	TYP3705	TYP3800	UDEVADD	UDEVBLOK	UDEVCLAS	UDEVFTR	UDEVMODE	UDEVNCYL	UDEVRELN	UDEVSTAT	UDEVTDSK
	UDEVTYPC	UDEVTYPE	UDEVVRR	UDEV3158	VCHADD	VCHBLOK	VCHBMX	VCHCUTEL	VCHSEL	VCHSIZE	VCHTYPE	VCONSIZE	VCUADD
	VCUBLOK	VCUCTCA	VCUDVTBL	VCUSHRD	VCUSIZE	VCUTYPE	VDEVADD	VDEVBLOK	VDEVEND	VDEVCLAS	VDEVCON	VDEVCOPY	VDEVCSPL
	VDEVDED	VDEVEOF	VDEVEXTN	VDEVFLAG	VDEVFLG2	VDEVLINK	VDEVNRDY	VDEVRDO	VDEVREAL	VDEVRELN	VDEVRRB	VDEVRRF	VDEVRSRL
	VDEVSFLG	VDEVSIZE	VDEVSTAT	VDEVTDSK	VDEVTERM	VDEVTMAT	VDEVTYPC	VDEVTYPE	VDEVUSER	VMBLOK	VMCHCNT	VMCHSTRT	VMCHTBL
	VMCUCNT	VMCUSTRT	VMDIST	VMDVCNT	VMDVSTRT	VMFBMX	VMFSTAT	VMOSTAT	VMSYSCP	VMTERM	VMVTERM	VRRBLOK	VRRSIZE
	VSPXBLOK	VSPXDIST	VSPXLEN	VSPXSIZE	ZEROES								
DMKVER	ADSPCH	AFREE	AFRET	ALARM	APTRAN	AQCNWT	BRING	CLASDASD	CPUID	C1	DDRCUA1	DDRCUA2	DDRKEYN
	DDRREC	DDRSIZE	DEFER	DMKCVTBH	DMKFREE	DMKFRET	DMKIOEVR	DMKPTRAN	DMKQCNWT	DMKSCNRD	DMKSCNVU	DMKVATRN	EXTMODE
	FTR2311B	FTR2311T	FTR35MB	F1	F15	F24	F256	F4	F4095	F7	F8	LOCK	MDRCUA1
	MDRKEYN	MDRREC	MDRSENS	MDRVOL	MIHCUA1	MIHKEYN	MIHREC	MIHSIZE	MIHVOL	MRDSIZE	NORET	OBRCPIDN	OBRCUA
	OBRCUAIN	OBRCUAPR	OBRHAN	OBRHSIZE	OBRKEYN	OBRLSIZE	OBRLSKN	OBRPGMN	OBRRECN	OBRSENSN	OBRSSIZE	OBRSWSN	OBRVOLN
	OBR2SIZE	OBR3SIZE	OBR33SNS	OPERATOR	PSA	RDEVBLOK	RDEVDED	RDEVFTR	RDEVSER	RDEVSTAT	RDEVTYPC	RDEVTYPE	R0
	R1	R10	R11	R12	R13	R14	R15	R2	R3	R4	R5	R6	R7
	R8	R9	SAVEAREA	SAVEREGS	SAVERETN	SAVER12	SAVEWRK1	SAVEWRK2	SAVEWRK3	SAVEWRK4	SAVEWRK5	SAVEWRK7	SAVEWRK8
	SAVEWRK9	TRANMODE	TYP2305	TYP2314	TYP3330	TYP3340	TYP3350	VDEVBLCK	VDEVDED	VDEVREAL	VDEVRELN	VDEVSTAT	VDEVTYPC
	VDEVTYPE	VMBLOK	VMEXWAIT	VMGPRS	VMPSW	VMRSTAT	VMSEG	VMSTOR	VMUSER				
DMKVIO	ADSPCH	AFRET	APTRAN	ATTN	AVMREAL	BRING	BUSY	CCC	CDC	CE	CLASDASD	CLASGRAF	CLASSPEC
	CLASTERM	CLASURI	CLASURO	CSW	CUE	C1	DE	DEFER	DMKCCHRF	DMKDSPCH	DMKFRET	DMKPTRAN	DMKPTRUL
	DMKSCNVU	DMKTRCSW	DMKTRDSI	DMKTRDWT	DMKTRKFP	DMKUNTFR	DMKUNTRN	F3	F8	IFCC	IL	INTREQ	IOBALTSK
	IOBCAW	IOBCC2	IOBCC3	IOBCSW	IOBFATAL	IOBFLAG	IOBHIC	ICBIOER	IOBLINK	IOBLOK	IOBMISC	IOBMISC2	IOBRELCU

MODULE	EXTERNAL REFERENCES (LABELS AND MODULES)												
	IOBSIOF	IOBSIZE	IOBSPEC	IOBSPEC2	IOBSTAT	IOBTIO	IOBUNSL	IOBVADD	IOBWRAP	IOERBLOK	IOERCSW	IOERDATA	IOEREXT
	IOERSIZE	LOCK	PCI	PREFIXA	PSA	R0	R1	R10	R11	R12	R14	R15	R2
	R3	R4	R5	R6	R7	R8	TEMPSAVE	TREXCSW	TREXCTL2	TREXT	TYP3340	TYP3704	UC
	VCHADD	VCHBLOK	VCHBMX	VCHBUSY	VCHCEDEV	VCHCEPND	VCHCUINT	VCHCUTBL	VCHSEL	VCHSTAT	VCHTYPE	VCUACTV	VCUADD
	VCUBLOK	VCUBUSY	VCUCEPND	VCUCHBSY	VCUCTCA	VCUCUEPN	VCUDVINT	VCUDVTBL	VCUINTS	VCUSHRD	VCUSTAT	VCUTYPE	VDEVADD
	VDEVBLOK	VDEVBUSY	VDEVCHAN	VDEVCHBS	VDEVCSW	VDEVCUE	VDEVDED	VDEVFLAG	VDEVINTS	VDEVIOB	VDEVIOER	VDEVNRDY	VDEVPEND
	VDEVPOSN	VDEVPOST	VDEVRELN	VDEVSAS	VDEVSTAT	VDEVTYPC	VDEVTYPE	VDEVUC	VMBLOK	VMCHSTRT	VMCUSTRT	VMDSTAT	VMDVSTRT
	VMESTAT	VMEXTCM	VMEXWAIT	VMIDLE	VMIOACTV	VMIOINT	VMIOPND	VMIOWAIT	VMPEND	VMPRIDSP	VMPSW	VMQSTAT	VMRSTAT
	VMSEG	VMTIO	VMTRBRIN	VMTRCTL	VMTREXT	VMTRIO	VMTRSIO	XTNDLOCK					
DMKVMA	ACORETBL	ADSPCH	AFREE	AFRET	APSTAT1	APSTAT2	APUOPER	BALRSAVE	BALR0	BALR2	CORCFLCK	CORFLAG	CORFLUSH
	CORFREE	CORIOLCK	CORPGPNT	CORRSV	CORSWPNT	CORTABLE	CORVM	CPEXADD	CPEXBLOK	CPEXREGS	CPEXSIZE	CPPTLBR	DMKCFMBK
	DMKCVTBH	DMKDSPCH	DMKDSPNP	DMKERMSG	DMKFREE	DMKFRET	DMKLOKDF	DMKLOKSY	DMKPTRFT	DMKPTRRC	DMKPTRSC	DMKPTRUL	DMKSTKCP
	DMKSTKMP	F1	F16	F256	F4095	F8	HALFPAGE	LOCK	LOCKSAV	LPUADDR	PAGACT	PAGCORE	PAGINVAL
	PAGTABLE	PROCIO	PSA	R0	R1	R10	R11	R12	R13	R14	R15	R2	R3
	R4	R5	R6	R7	R8	R9	SAVE	SAVEAREA	SEGINV	SEGPAGE	SEGPLEN	SEGTABLE	SHRNAME
	SHRSEGCT	SHRSEGNM	SHRTABLE	SWPFLAG	SWPTABLE	SWPTRANS	SWPVPAGE	SWTHSAVE	TEMPR0	TEMPR2	TEMPR5	TEMPR6	TYPE
	VMABLOK	VMAFPNT	VMASHRBK	VMASSIST	VMBLOK	VMLOGOFF	VMPAGES	VMRSTAT	VMSEG	VMSHRPRC	XPAGNUM		
DMKVMC	ADSPCH	AFREE	AFRET	APSTAT1	APTRAN	APTRLK	APUOPER	ASYSVM	BRING	CPEXADD	CPEXBLOK	CPEXR11	CPEXR12
	CPEXSIZE	C1	DEFER	DMKDSPCH	DMKFREE	DMKFRET	DMKLOKSW	DMKPSAFC	DMKPSASC	DMKPSASP	DMKPTRAN	DMKPTRUL	DMKSCHDL
	DMKSCNAU	DMKSTKCP	ECBLOK	EXTCR0	F1	F7	LOCK	PSA	R0	R1	R10	R11	R12
	R13	R14	R15	R2	R3	R4	R5	R6	R7	R8	R9	SAVEAREA	SAVEREGS
	SAVER11	SAVER2	SAVER5	SAVER6	SAVEWRK1	SAVEWRK5	SAVEWRK6	SAVEWRK7	SYSTEMID	TIMEDISP	VMBCAUTH	VMBLOK	VMCAAUTS
	VMCACNT	VMCAPRTY	VMCAQIES	VMCASTAT	VMCBLOK	VMCBSIZE	VMCCBUSY	VMCCRECP	VMCCSTAT	VMCCXINT	VMCEFLG	VMCFPNT	VMCFUNC
	VMCKEY	VMCLENA	VMCMID	VMCMLEN	VMCPARM	VMCPAUTS	VMCPFLG1	VMCPFUNC	VMCPIDEN	VMCPLEN	VMCPMID	VMCPNT	VMCPPRTY
	VMCPRTY	VMCPSENR	VMCPSENX	VMCPSMSG	VMCPUSE'	VMCPUSER	VMCPVADA	VMCPVADB	VMCRESP	VMCRJCT	VMCSMAX	VMCSTAT	VMCTOD
	VMCUSE	VMCUSER	VMCVADA	VMCVADB	VMCXCODE	VMCXMASK	VMCXSTAT	VMC01	VMC02	VMC03	VMC04	VMC05	VMC06
	VMC07	VMC08	VMC09	VMC10	VMC11	VMC12	VMC13	VMC14	VMC15	VMC16	VMC17	VMC18	VMC19
	VMC20	VMEXTPND	VMEXWAIT	VMPSTAT	VMPSW	VMPXINT	VMRSTAT	VMSEG	VMSPMFLG	VMSPMON	VMUSER	VMVCR0	VMV370R
	XINTBLOK	XINTCODE	XINTNEXT	XINTSIZE	XINTSORT	XPAGNUM							
DMKVMI	ATTN	BUSY	CAW	CC	CD	CE	CLASDASD	CLASSPEC	CLASTAPE	CLASURI	CSW	DE	EXTMODE
	FLAG1	FLAG2	IL	INTTIO	IPLADDR	IPLCCW1	IPLPSW	LOCK	PSA	R0	R1	R10	R11
	R12	R13	R14	R15	R2	R3	R4	R5	R6	R7	R9	SAVEREGS	SEARCH
	SEEK	SENSE	SILI	SKIP	SM	TAPE	TIC	TYPCTCA	TYPE	TYPRDR	TYPUNSUP	TYP2401	TYP2415
	TYP2420	TYP2501	TYP2540R	UC	UE	VMMCODE	VMMTEXT						
DMKVSI	ADSPCH	AFREE	AFRET	APSTAT1	APTRAN	APUOPER	ASYSVM	ATTN	AVMREAL	BALRSAVE	BALR8	BLKMPX	BRING
	BUSY	CAW	CC	CCC	CD	CDC	CE	CHBM370	CHXBLOK	CHXFLAG	CLASDASD	CLASGRAF	CLASSPEC
	CLASTAPE	CLASTERM	CLASURI	CLASURO	CODE	CPEXADD	CPEXBLOK	CPEXMISC	CPEXREGS	CPEXSIZE	CSW	CUE	C1
	DE	DEFER	DMKCCHRF	DMKCCWTR	DMKDSPA	DMKDSPCH	DMKDSERQ	DMKDSPRU	DMKFREE	DMKFRET	DMKIOSQV	DMKLOKDF	DMKLOKSY
	DMKPTRAN	DMKSCHDL	DMKSCNVU	DMKSSSMQ	DMKSTKDE	DMKSTKIO	DMKSTKMP	DMKSTKOP	DMKTRDSI	DMKVCASH	DMKVCAST	DMKVCATS	DMKVCNEX
	DMKVIOC1	DMKVIOIN	DMKVIOMK	DMKVIOXK	DMKVSPEX	DMKVSPTO	FTR35MB	FTR70MB	F1	F10	F15	F240	F4
	F4095	F4096	F8	IDA	IFCC	IOBCAW	IOBCC3	IOBCLN	IOBCSW	IOBFLAG	IOBFPNT	IOBHIO	IOBIRA
	IOBLINK	IOBLOK	IOBMISC	IOBRADD	IOBRCAW	IOBRELCU	IOBSICF	IOBSIZE	IOBSPEC	IOBSPEC2	IOBSTAT	IOBTIO	IOBUSER

MODULE	EXTERNAL REFERENCES (LABELS AND MODULES)												
	IOBVADD	IOERBLOK	IOEREXT	IOERSIZE	LOCK	LOCKSAV	LPUADDR	MSSNEXT	MSSPRES	MSSTASK1	MSSTASK3	MSSUSER	PCI
	PREFIXA	PROCIO	PSA	PSAMSS	RDEVAIOB	RDEVALT	RDEVBLCK	RDEVFTR	RDEVMDL	RDEVSTA2	RDEVTYPC	R0	R1
	R10	R11	R12	R13	R14	R15	R2	R3	R4	R5	R6	R7	R8
	R9	SAVE	SKIP	SM	SYSVIRT	TEMPSAVE	TIMEDISP	TRACBEF	TRACCURR	TRACEND	TRACFLG2	TRACPROC	TRACSTRT
	TRAC0D	TRCCLCH	TRCCSW	TRCHALT	TYPCTCA	TYPE	TYPRDR	TYP2314	TYP3210	TYP3330	TYP3340	TYP3350	TYP3705
	UC	UE	VCHADD	VCHBLOK	VCHBMX	VCHBUSY	VCHCEPND	VCHCUINT	VCHCUTBL	VCHDED	VCHSEL	VCHSTAT	VCHTYPE
	VCUACTV	VCUADD	VCUBLOK	VCUBUSY	VCUCTCA	VCUCUEPN	VCUDVINT	VCUDVTBL	VCUINTS	VCUSHRD	VCUSTAT	VCUTYPE	VDEVADD
	VDEVBLOK	VDEVBND	VDEVBUSY	VDEVCHAN	VDEVCHBS	VDEVCSW	VDEVCUE	VDEVDED	VDEVDIAL	VDEVENAB	VDEVFLAG	VDEVFLG2	VDEVINTS
	VDEVIOB	VDEVIOER	VDEVNRDY	VDEVODE	VDEVPEND	VDEVPOST	VDEVRDO	VDEVREAL	VDEVRES	VDEVRRB	VDEVRRF	VDEVSPL	VDEVSTAT
	VDEVTYPC	VDEVTYPE	VDEVUC	VIRTUAL	VMACTDEV	VMBLOK	VMCHSTRT	VMCUSTRT	VMDFTPNT	VMDSTAT	VMDVSTRT	VMECEXT	VMESTAT
	VMEXTCM	VMEXWAIT	VMGPRS	VMIDLE	VMINST	VMIOACTV	VMIOCNT	VMIOINT	VMICPND	VMIOWAIT	VMNOTRAN	VMPEND	VMPSTAT
	VMPSW	VMRSTAT	VMSEG	VMTIO	VMTRCTL	VMTRSIO	VMUSER	VMVCR0	VMV370R	VRRBLOK	VRRRES	VRRSTAT	XRIGHT16
DMKVSP	ADDSFB	ADSPCH	AFREE	AFRET	APTRAN	ARSPRD	ASYSVM	BRING	CC	CD	CE	CHGSFB	CLASURI
	CLASURO	CMDREJ	CPEXADD	CPEXBLOK	CPEXFPNT	CPEXR1	CPEXR11	CPEXR8	CPEXSIZE	CSW	C1	DATACHK	DE
	DEFER	DMKCKSPL	DMKCVTBH	DMKCVTDT	DMKDSPCH	DMKERMSG	DMKFREE	DMKFRET	DMKPGTSG	DMKPGTVG	DMKPGTVR	DMKPSACC	DMKPSASC
	DMKPTRAN	DMKRPAGT	DMKRPAPT	DMKSCNVD	DMKSCNVU	DMKSPLCV	DMKSPLDL	DMKSPLOV	DMKSTKCP	DMKTMRPT	DMKVIOMK	DMKVMASH	DMKVSQPD
	FFS	F0	F1	F4	F4095	F4096	IDA	IL	INTREQ	IOERETN	LOCK	OPNSFB	PCI
	PCIF	PRGC	PRTC	PRTCHN	PSA	READ	READER1	RECBLOK	RECCYL	RECMAP	RECPNT	RECSIZE	RECUSED
	R0	R1	R10	R11	R12	R13	R14	R15	R2	R3	R4	R5	R6
	R7	R8	R9	SAVEAREA	SAVEREGS	SAVER0	SAVER1	SAVER2	SAVER8	SAVEWRK2	SAVEWRK6	SENSE	SFBCLAS
	SFBDUMP	SFBEOF	SFBFILID	SFBFLAG	SFBFLAG2	SFBFLNMT	SFBHOLD	SFBINUSE	SFBLAST	SFBLOK	SFBMON	SFBNOHLD	SFBOPEN
	SFBPNT	SFBPURGE	SFBRECER	SFBRECNO	SFBRECS	SFBRECSZ	SFBSTART	SFBTIME	SFBUHOLD	SFBUSER	SILI	SKIP	SPFILID
	SPLINK	SPNXTPAG	SPPREPAG	SPRECNUM	SPSIZE	SPTIME	SYSTEM	TEMPR0	TEMPR1	TIC	TYPPRT	TYPPUN	TYPTIMER
	TYP1052	TYP3203	TYP3210	TYP3211	TYP3505	UC	UE	VCHADD	VCHBLCK	VCHBMX	VCHCEDEV	VCHCEPND	VCHCUINT
	VCHCUTBL	VCHSEL	VCHSTAT	VCHTYPE	VCUADD	VCUBLOK	VCUDVINT	VCUDVTBL	VDEVADD	VDEVBLOK	VDEVBUSY	VDEVCCW1	VDEVCFCL
	VDEVCHAN	VDEVCHBS	VDEVCLAS	VDEVCONT	VDEVCSPL	VDEVCSW	VDEVDED	VDEVDIAG	VDEVDLY	VDEVEOF	VDEVFCBK	VDEVFEED	VDEVFLAG
	VDEVHOLD	VDEVINTS	VDEVIOCT	VDEVKEY	VDEVNRDY	VDEVPEND	VDEVPURG	VDEVSFLG	VDEVSNSE	VDEVSPL	VDEVSTAT	VDEVSVC	VDEVTYPC
	VDEVTYPE	VDEVUNIT	VFCBBLOK	VFCBCHL	VFCBCNT	VFCBEOF	VFCBFLAG	VFCBLOAD	VFCBNDEX	VFCBSIZE	VMBLOK	VMCHSTRT	VMCHTBL
	VMCRDS	VMCUSTRT	VMDVSTRT	VMESTAT	VMEXTCM	VMEXWAIT	VMINST	VMIOINT	VMIOPND	VMLINS	VMPEND	VMPNCH	VMPSW
	VMRSTAT	VMSEG	VMTTIME	VMUSER	VSPBUFBK	VSPBUFSZ	VSPCAW	VSPCCW	VSPDPAGE	VSPIDACT	VSPIDAL	VSPIDASW	VSPIDAW2
	VSPLCTL	VSPNEXT	VSPRECNO	VSPSFBLK	VSPSIZE	VSPVPAGE	ZEROES						
DMKVSQ	AFREE	AFRET	APTRAN	APTRLK	ASYSVM	BRING	CC	C1	DEFER	DMKBOXHR	DMKFREE	DMKFRET	DMKPGTSG
	DMKPGTVG	DMKPGTVR	DMKPTRAN	DMKPTRUL	DMKRPAGT	DMKRPAPT	DMKSPLCV	DMKSPLDL	DMKVSP	DMKVSPWA	FFS	F1	F4096
	F8	IOERETN	LOCK	PSA	R0	R1	R10	R11	R12	R14	R15	R2	R3
	R4	R5	R6	R7	R8	R9	SFBCLAS	SFBFILID	SFBFLAG	SFBFLAG2	SFBFLNMT	SFBLAST	SFBLOK
	SFBMISC1	SFBPNT	SFBPURGE	SFBRECER	SFBRECNO	SFBSTART	SFBTIME	SFBTYPE	SILI	SPFILID	SPLINK	SPNXTPAG	SPPREPAG
	SPRECNUM	SPSIZE	SPTIME	SYSTEM	TYPPRT	TYP3210	VDEVBLOK	VDEVCFCL	VDEVCLAS	VDEVPURG	VDEVSFLG	VDEVSPL	VDEVTYPE
	VMBLOK	VMDVSTRT	VMOSTAT	VMSEG	VMSYSOP	VSPBUFBK	VSPBUFSZ	VSPDPAGE	VSPLCTL	VSPNEXT	VSPSFBLK	VSPSIZE	VSPVPAGE
DMKWRM	ACNTBLOK	ACNTCCW	ACNTDATA	ACNTNEXT	ACNTSIZE	ADDSFB	AFREE	ALARM	APTRAN	APTRLK	AQCNWT	ARIODV	ASYSVM
	BLK	BRING	BUFFER	CC	CHGSHQ	CKPBLOK	CKPNAME	CKPRMAX	CKPSIZE	CLASSPEC	CLASTERM	C1	DEFER
	DMKCKSIN	DMKCKSPL	DMKCKSWM	DMKCVTBD	DMKERMSG	DMKFREE	DMKPGTVG	DMKPGTVR	DMKPTRAN	DMKQCNWT	DMKQNTBL	DMKRPAGT	DMKRPAPT
	DMKRSPAC	DMKRSPCV	DMKRSPDL	DMKRSPHQ	DMKRSPID	DMKRSPPR	DMKRSPPU	DMKRSPRD	DMKSCNRU	DMKSYSDT	DMKSYSLG	DMKSYSOW	DMKSYSWM
	FFS	F0	F256	F8	LOCK	NICBLOK	NICDISA	NICENAE	NICFLAG	NICLGRP	NICSIZE	NICSTAT	NICTERM

MODULE EXTERNAL REFERENCES (LABELS AND MODULES)

NICTYPE	NPRCNT	NPRNAME	NPRPNT	NPRTBL	OPERATOR	OWNDLIST	OWNDRDEV	PCHCHN	POINTER	PRTCHN	PSA	RDEVALLN
RDEVAUTO	RDEVBLOK	RDEVCKPT	RDEVCLAS	RIEVCODE	RDEVDISA	RDEVDRAN	RDEVENAB	RDEVFLAG	RDEVFSEP	RDEVIMAG	RDEVMAX	RDEVNCP
RDEVNICL	RDEVRECS	RDEVSEP	RDEVSER	RDEVSPL	RDEVSTAT	RDEVTYPC	RDEVTYPE	RDEVXSEP	RDRCHN	RECBLOK	RECCYL	RECPNT
RECSIZE	RECUSED	R0	R1	R10	R11	R12	R13	R14	R15	R2	R3	R4
R5	R6	R7	R8	R9	SAVEAREA	SAVEREGS	SAVER2	SAVEWRK1	SAVEWRK2	SAVEWRK3	SAVEWRK4	SAVEWRK6
SAVEWRK7	SAVEWRK8	SFBDATE	SFBEOF	SFBFILID	SFBFLAG	SFBFLAG2	SFBFNAME	SFBFTYPE	SFBINUSE	SFBLOK	SFBOPEN	SFBPNT
SFBRECER	SFBRECS	SFBRSTRT	SFBSIZE	SHQBLOK	SHQBSIZE	SILI	STARTIME	SYSIPLDV	SYSTEM	TYPBSC	TYPE	TYP2305
TYP3330	TYP3340	TYP3350	TYP3705	TYP3800	VMBLOK	VMSEG	ZEROES					

LABEL	COUNT	REFERENCES											
ABORT	000003	DMKNLD	DMKNLE	DMKRNH									
ACCTACNO	000003	DMKHVD											
ACCTBLOK	000005	DMKACO	DMKCKP	DMKHVD	DMKSPL								
ACCTDIST	000002	DMKHVD	DMKSPL										
ACCTLENG	000004	DMKHVD	DMKUSO										
ACCTUSER	000004	DMKACO	DMKCKP	DMKHVD	DMKSPL								
ACNTBACK	000007	DMKACO	DMKRSE										
ACNTBLOK	000024	DMKACO	DMKCKP	DMKHVD	DMKJRL	DMKRSE	DMKWRM						
ACNTCCW	000009	DMKACO	DMKCKP	DMKWRM									
ACNTCODE	000007	DMKACO	DMKCKP	DMKHVD									
ACNTCONT	000002	DMKACO	DMKCKP										
ACNTDATA	000014	DMKACO	DMKCKP	DMKHVD	DMKWRM								
ACNTDEVC	000004	DMKACO	DMKCKP										
ACNTIOCT	000002	DMKACO	DMKCKP										
ACNTNCYL	000002	DMKACO	DMKCKP										
ACNTNEXT	000014	DMKACO	DMKCKP	DMKWRM									
ACNTNUM	000003	DMKACO	DMKCKP	DMKHVD									
ACNTPGRD	000002	DMKACO	DMKCKP										
ACNTSIZE	000011	DMKACO	DMKHVD	DMKJRL	DMKWRM								
ACNTSTOP	000018	DMKACO	DMKCKP										
ACNTTIME	000002	DMKACO	DMKCKP										
ACNTUSER	000005	DMKACO	DMKCKP	DMKHVD									
ACNTVTIM	000002	DMKACO	DMKCKP										
ACOACCL	000001	DMKACO											
ACOACCM	000002	DMKACO											
ACOCHK	000001	DMKACO											
ACOEXIT	000005	DMKACO											
ACORETBL	000092	DMKACO	DMKATS	DMKBLD	DMKCCW	DMKCDS	DMKCFC	DMKCPI	DMKCPU	DMKCPV	DMKCSB	DMKDGD	DMKFRE
		DMKMCC	DMKMCH	DMKMNI	DMKPAG	DMKPGS	DMKPSA	DMKPTR	DMKRPA	DMKSPL	DMKUDR	DMKUDU	DMKUNT
		DMKVMA											
ACORETN	000002	DMKACO											
ACTIVTRQ	000015	DMKAPI	DMKCPU	DMKDSP	DMKPSA	DMKSCH							
ACTSFB	000005	DMKCKS											
ADDSFB	000017	DMKCKS	DMKMIA	DMKNLE	DMKSPL	DMKVSP	DMKWRM						
ADSPCH	000152	DMKACO	DMKALG	DMKCCW	DMKCFM	DMKCFO	DMKCFP	DMKCNS	DMKCPB	DMKCPI	DMKCPS	DMKCPU	DMKCSB
		DMKDGD	DMKDIA	DMKDSB	DMKDSP	DMKENT	DMKEXT	DMKGIO	DMKGRF	DMKHVC	DMKIOE	DMKIOG	DMKIOS
		DMKLOC	DMKLOG	DMKLOK	DMKMCD	DMKMCH	DMKMCT	DMKMNI	DMKMCN	DMKNLD	DMKNLE	DMKPAG	DMKPGS
		DMKPGT	DMKPRG	DMKPRV	DMKPSA	DMKPTR	DMKQCN	DMKRGA	DMKRGB	DMKRNH	DMKRSE	DMKRSP	DMKSCH
		DMKSEP	DMKSPL	DMKSSS	DMKSVC	DMKTAP	DMKTCS	DMKTDK	DMKTMR	DMKTRD	DMKTRK	DMKUDR	DMKUNT
		DMKUSO	DMKVAT	DMKVCA	DMKVCN	DMKVDC	DMKVDE	DMKVER	DMKVIO	DMKVMA	DMKVMC	DMKVSI	DMKVSP
ADTRANS	000010	DMKPRV											
AEXTSP	000034	DMKACO	DMKCLK	DMKCPI	DMKCPS	DMKCPU	DMKDSP	DMKFRE	DMKIOS	DMKMCH	DMKMCT	DMKPTR	DMKSCH
AFREE	000450	DMKACO	DMKALG	DMKAPI	DMKATS	DMKBLD	DMKBSC	DMKCCH	DMKCCW	DMKCDE	DMKCDM	DMKCDS	DMKCFC
		DMKCFD	DMKCFG	DMKCFH	DMKCFM	DMKCFO	DMKCFP	DMKCFS	DMKCKS	DMKCNS	DMKCPB	DMKCPI	DMKCPS
		DMKCPU	DMKCPV	DMKCQG	DMKCQH	DMKCQP	DMKCQR	DMKCQY	DMKCSE	DMKCSO	DMKCSP	DMKCSQ	DMKCST
		DMKCSU	DMKCSV	DMKDAS	DMKDEF	DMKDGD	DMKDIA	DMKDIB	DMKDRD	DMKDSB	DMKDSP	DMKERM	DMKEXT

LABEL	COUNT	REFERENCES											
		DMKFRE	DMKGIO	DMKGRF	DMKGRT	DMKHVC	DMKHVD	DMKIOE	DMKIOF	DMKIOG	DMKIOS	DMKISM	DMKJRL
		DMKLD00E	DMKLNK	DMKLOC	DMKLOG	DMKLOK	DMKMCC	DMKMCD	DMKMCH	DMKMCT	DMKMIA	DMKMID	DMKMNI
		DMKMON	DMKMSG	DMKMSW	DMKNES	DMKNET	DMKNLD	DMKNLE	DMKPAG	DMKPGS	DMKPGT	DMKPSA	DMKPTR
		DMKQCN	DMKRGA	DMKRGB	DMKRNH	DMKRPA	DMKRSE	DMKRSP	DMKSCH	DMKSPL	DMKSSS	DMKSVC	DMKTAP
		DMKTCS	DMKTDK	DMKTHI	DMKTMR	DMKTRA	DMKTRD	DMKTRK	DMKUDR	DMKUDU	DMKUNT	DMKUSO	DMKVAT
		DMKVCA	DMKVCH	DMKVCN	DMKVDA	DMKVDC	DMKVDD	DMKVDE	DMKVDR	DMKVDS	DMKVER	DMKVMA	DMKVMC
		DMKVSI	DMKVSP	DMKVSQ	DMKWRM								
AFRET	000433	DMKACO	DMKATS	DMKBLD	DMKBSC	DMKCCH	DMKCCW	DMKCDB	DMKCDM	DMKCDS	DMKCFD	DMKCFG	DMKCFH
		DMKCFM	DMKCFO	DMKCFP	DMKCFS	DMKCKS	DMKCNS	DMKCPB	DMKCPI	DMKCPS	DMKCPU	DMKCQG	DMKCQH
		DMKCQP	DMKCQR	DMKCQY	DMKCSB	DMKCSO	DMKCSP	DMKCSQ	DMKCST	DMKCSU	DMKCSV	DMKDAS	DMKDEF
		DMKDGD	DMKDIA	DMKDRD	DMKDSB	DMKDSP	DMKERM	DMKFRE	DMKGIO	DMKGRF	DMKHVC	DMKHVD	DMKIOE
		DMKIOF	DMKIOG	DMKIOS	DMKJRL	DMKLD00E	DMKLNK	DMKLOC	DMKLOG	DMKLOH	DMKMCC	DMKMCD	DMKMIA
		DMKMNI	DMKMON	DMKMSG	DMKMSW	DMKNES	DMKNET	DMKNLD	DMKNLE	DMKPAG	DMKPGS	DMKPGT	DMKPTR
		DMKQCN	DMKRGA	DMKRGB	DMKRNH	DMKRSE	DMKRSP	DMKSCH	DMKSEP	DMKSPL	DMKSSS	DMKSVC	DMKTAP
		DMKTCS	DMKTDK	DMKTHI	DMKTMR	DMKTRA	DMKTRC	DMKTRD	DMKTRK	DMKUDR	DMKUDU	DMKUNT	DMKUSO
		DMKVAT	DMKVCA	DMKVCH	DMKVCN	DMKVDA	DMKVDC	DMKVDD	DMKVDE	DMKVDR	DMKVDS	DMKVER	DMKVIO
		DMKVMA	DMKVMC	DMKVSI	DMKVSP	DMKVSQ							
ALARM	000091	DMKACO	DMKCCH	DMKCKP	DMKCLK	DMKCNS	DMKCPI	DMKCQP	DMKDAS	DMKDMP	DMKDSB	DMKERM	DMKGRF
		DMKJRL	DMKMCH	DMKMCT	DMKMID	DMKMSG	DMKMSW	DMKOPR	DMKPGT	DMKQCN	DMKRGB	DMKRNH	DMKRSP
		DMKSAV	DMKUDR	DMKVCN	DMKVER	DMKWRM							
ALOCBLOK	000016	DMKCPI	DMKMON	DMKPGT	DMKTDK	DMKVDC							
ALOCCYL1	000006	DMKCPI	DMKTDK	DMKVDC									
ALOCCYL2	000005	DMKCPI	DMKTDK	DMKVDC									
ALOCMAP	000016	DMKCPI	DMKPGT	DMKTDK	DMKVDC								
ALOCMAX	000015	DMKCPI	DMKMON	DMKPGT	DMKVDC								
ALOCNTMP	000007	DMKCPI	DMKMON	DMKVDC									
ALOCPNT	000004	DMKCPI	DMKTDK	DMKVDC									
ALOCUSED	000010	DMKCPI	DMKMON	DMKPGT	DMKVDC								
ALOKSP	000020	DMKDSP	DMKFRE	DMKIOS	DMKPAG	DMKPSA	DMKSCH	DMKSTK					
ALOKVM	000005	DMKAPI	DMKCPU	DMKDSP	DMKMCH								
AMCHAREA	000011	DMKCCH	DMKCFO	DMKCPU	DMKIOG	DMKMCH							
APAGCP	000005	DMKCPI	DMKCPV	DMKSVC									
APOINT	000001	DMKLD00E											
APSTAT1	000519	DMKACO	DMKALG	DMKAPI	DMKATS	DMKBLD	DMKBSC	DMKCDB	DMKCDM	DMKCDS	DMKCFG	DMKCFM	DMKCFO
		DMKCFP	DMKCFS	DMKCFT	DMKCKS	DMKCLK	DMKCNS	DMKCPI	DMKCPS	DMKCPU	DMKCPV	DMKCQP	DMKCQR
		DMKCQY	DMKCSO	DMKCSV	DMKDAS	DMKDIA	DMKDSB	DMKDSP	DMKENT	DMKEXT	DMKFRE	DMKGRF	DMKHVD
		DMKIOE	DMKIOG	DMKIOS	DMKJRL	DMKLOG	DMKLOH	DMKLOK	DMKMCD	DMKMCH	DMKMCT	DMKMIA	DMKMID
		DMKMNI	DMKMON	DMKMSG	DMKMSW	DMKNES	DMKNET	DMKNLD	DMKNLE	DMKPAG	DMKPGS	DMKPGT	DMKPRG
		DMKPRV	DMKPSA	DMKPTR	DMKQCN	DMKRGA	DMKRGB	DMKRNH	DMKRSE	DMKRSP	DMKSCH	DMKSEP	DMKSPL
		DMKSTK	DMKSVC	DMKTAP	DMKTCS	DMKTHI	DMKTMR	DMKTRK	DMKTRM	DMKVAT	DMKVCA	DMKVCH	DMKVDA
		DMKVDD	DMKVDE	DMKVDR	DMKVDS	DMKVMA	DMKVMC	DMKVSI					
APSTAT2	000025	DMKATS	DMKBLD	DMKCPU	DMKDSP	DMKIOG	DMKMCH	DMKPGS	DMKPRV	DMKPTR	DMKRPA	DMKVAT	DMKVMA
APSTAT3	000004	DMKDSP	DMKEXT	DMKLOK									
APSTAT4	000034	DMKAPI	DMKCLK	DMKCPU	DMKDSP	DMKEXT	DMKLOK	DMKMCH	DMKMCT				
APSV	000006	DMKLD00E											
APTRAN	000173	DMKAPI	DMKBLD	DMKCCH	DMKCCW	DMKCDB	DMKCDM	DMKCDS	DMKCFD	DMKCFG	DMKCFH	DMKCKS	DMKCNS

LABEL	COUNT	REFERENCES											
		DMKCPB	DMKCPI	DMKCPS	DMKCPU	DMKCPV	DMKCSB	DMKCSO	DMKDGD	DMKDRD	DMKDSP	DMKERM	DMKGIO
		DMKGRT	DMKHVC	DMKHVD	DMKIOF	DMKMCC	DMKMIA	DMKMON	DMKPGS	DMKPRG	DMKPRV	DMKPTR	DMKQCN
		DMKRPA	DMKRSP	DMKSEP	DMKSNC	DMKSPL	DMKSSS	DMKSVC	DMKTCS	DMKTMR	DMKTRC	DMKTRD	DMKTRK
		DMKUDR	DMKUDU	DMKVAT	DMKVCH	DMKVCN	DMKVDR	DMKVER	DMKVIO	DMKVMC	DMKVSI	DMKVSP	DMKVSQ
		DMKWRM											
APTRLK	000046	DMKACO	DMKAPI	DMKATS	DMKCCW	DMKCFG	DMKCFH	DMKCKS	DMKCPI	DMKCPS	DMKCPU	DMKDGD	DMKIOG
		DMKMCC	DMKMIA	DMKMNI	DMKMON	DMKNLD	DMKNLE	DMKRSP	DMKSEP	DMKSNC	DMKSPL	DMKSVC	DMKTCS
		DMKTRC	DMKUSO	DMKVMC	DMKVSQ	DMKWRM							
APUNONLN	000001	DMKCPI											
APUOPER	000351	DMKACO	DMKALG	DMKAPI	DMKATS	DMKBLD	DMKCDB	DMKCDM	DMKCDS	DMKCFG	DMKCFO	DMKCFP	DMKCFS
		DMKCLK	DMKCNS	DMKCPI	DMKCPS	DMKCPU	DMKCPV	DMKCQP	DMKCQR	DMKCQY	DMKCSV	DMKDIA	DMKDSP
		DMKEXT	DMKFRE	DMKGRF	DMKIOS	DMKJRL	DMKLOG	DMKLOH	DMKLOK	DMKMCD	DMKMCH	DMKMCT	DMKMIA
		DMKMID	DMKMNI	DMKMON	DMKMSG	DMKMSW	DMKNES	DMKNLD	DMKPAG	DMKPGS	DMKPRG	DMKPRV	DMKPSA
		DMKPTR	DMKQCN	DMKRGA	DMKRGB	DMKRNH	DMKSCH	DMKSPL	DMKSTK	DMKSVC	DMKTCS	DMKTHI	DMKTMR
		DMKVAT	DMKVCA	DMKVCH	DMKVDA	DMKVDD	DMKVMA	DMKVMC	DMKVSI				
AQCNWT	000227	DMKACO	DMKAPI	DMKBLD	DMKCDB	DMKCDM	DMKCDS	DMKCFC	DMKCFD	DMKCFH	DMKCFM	DMKCFO	DMKCFS
		DMKCLK	DMKCPB	DMKCPI	DMKCPS	DMKCPU	DMKCPV	DMKCQG	DMKCQH	DMKCQP	DMKCQR	DMKCQY	DMKCSB
		DMKCSO	DMKCSU	DMKCSV	DMKDAS	DMKDEF	DMKDIA	DMKDSB	DMKERM	DMKJRL	DMKLNK	DMKLOH	DMKMCC
		DMKMCD	DMKMCH	DMKMCT	DMKMIA	DMKMID	DMKMNI	DMKMSG	DMKMSW	DMKNES	DMKNET	DMKNLD	DMKNLE
		DMKPGT	DMKPRG	DMKPTR	DMKQCN	DMKRNH	DMKRSP	DMKSPL	DMKSVC	DMKTHI	DMKTRA	DMKTRC	DMKTRD
		DMKUDR	DMKUSO	DMKVCA	DMKVCH	DMKVCN	DMKVDA	DMKVDD	DMKVDR	DMKVER	DMKWRM		
ARIOCC	000001	DMKCKP											
ARIOCH	000011	DMKCCH	DMKCKP	DMKCPI	DMKCPS	DMKCPV	DMKCQP	DMKGRF	DMKIOG	DMKMNI	DMKSCN	DMKSSP	
ARIOCT	000011	DMKCCH	DMKCKP	DMKCPI	DMKCPS	DMKCPV	DMKCQP	DMKIOG	DMKMNI	DMKSCN	DMKSSP		
ARIOCU	000015	DMKCCH	DMKCKP	DMKCPI	DMKCPS	DMKCPV	DMKCQP	DMKDIA	DMKMNI	DMKNES	DMKSCN	DMKSSP	DMKVCH
ARIODC	000004	DMKLOG	DMKSCN	DMKSSS									
ARIODV	000054	DMKACO	DMKATS	DMKCCH	DMKCKP	DMKCKS	DMKCPI	DMKCPS	DMKCPU	DMKCPV	DMKCQP	DMKCQR	DMKCSO
		DMKDIA	DMKDRD	DMKGRF	DMKLOG	DMKMNI	DMKMON	DMKNES	DMKNET	DMKPAG	DMKPGS	DMKPGT	DMKPTR
		DMKSCN	DMKSPL	DMKSSP	DMKSSS	DMKUDR	DMKVCH	DMKVDC	DMKWRM				
ARIOPR	000005	DMKCKP	DMKCSB	DMKCSO	DMKSPL								
ARIOPU	000009	DMKACO	DMKCKP	DMKCSB	DMKCSO	DMKSPL							
ARIORD	000005	DMKCKP	DMKCSB	DMKCSO									
ARSPAC	000003	DMKACO											
ARSPPR	000012	DMKCKP	DMKCKS	DMKCQH	DMKCQR	DMKCSP	DMKCSQ	DMKCSU	DMKCSV	DMKSPL	DMKUSO		
ARSPPU	000011	DMKCKS	DMKCQH	DMKCQR	DMKCSP	DMKCSQ	DMKCSU	DMKCSV	DMKSPL	DMKUSO			
ARSPRD	000028	DMKCKS	DMKCQH	DMKCQR	DMKCSP	DMKCSQ	DMKCST	DMKCSU	DMKCSV	DMKDRD	DMKMIA	DMKNLE	DMKSPL
		DMKUSO	DMKVSP										
ASYSLC	000025	DMKACO	DMKBLD	DMKCFO	DMKCFT	DMKCKP	DMKLOC	DMKLOG	DMKLOH	DMKUDR	DMKUDU	DMKUSO	
ASYSOP	000026	DMKAPI	DMKCLK	DMKCPI	DMKCPS	DMKDAS	DMKJRL	DMKLOG	DMKMIA	DMKMID	DMKMSG	DMKMSW	DMKPSA
		DMKQCN	DMKUDU	DMKUSO	DMKVCH								
ASYSVM	000253	DMKACO	DMKAPI	DMKATS	DMKBLD	DMKCFG	DMKCFH	DMKCFP	DMKCKP	DMKCKS	DMKCNS	DMKCPB	DMKCPI
		DMKCPS	DMKCPU	DMKCPV	DMKCQP	DMKCSB	DMKCSC	DMKDIA	DMKDRD	DMKDSB	DMKDSP	DMKERM	DMKEXT
		DMKFRE	DMKGRF	DMKGRT	DMKHVC	DMKHVD	DMKIOE	DMKIOG	DMKIOS	DMKLOG	DMKLOK	DMKMCC	DMKMCH
		DMKMCT	DMKMIA	DMKMID	DMKMNI	DMKMON	DMKMSG	DMKNES	DMKNET	DMKNLD	DMKNLE	DMKPAG	DMKPGS
		DMKPGT	DMKPSA	DMKPTR	DMKRGA	DMKRGB	DMKRNH	DMKRSP	DMKSCN	DMKSEP	DMKSNC	DMKSPL	DMKSVC
		DMKTAP	DMKTCS	DMKTHI	DMKUDR	DMKUDU	DMKUSC	DMKVAT	DMKVCH	DMKVDA	DMKVDC	DMKVDD	DMKVDR

LABEL	COUNT	REFERENCES											
		DMKVMC	DMKVSI	DMKVSP	DMKVSQ	DMKWRM							
ATMRSN	000008	DMKCPU	DMKMON	DMKSCH	DMKTMR								
ATTN	000065	DMKCFC	DMKCFM	DMKCKP	DMKCNS	DMKCQR	DMKDDB	DMKDIR	DMKDMP	DMKDSP	DMKFMT	DMKGRF	DMKIOS
		DMKNLD	DMKNLE	DMKRNH	DMKRSE	DMKSSP	DMKSSS	DMKVCA	DMKVIO	DMKVMI	DMKVSI		
ATTR2	000037	DMKGRF	DMKGRT	DMKGRW	DMKRGA	DMKRGB							
ATTR457	000005	DMKGRF	DMKGRT	DMKGRW	DMKRGA	DMKRGB							
ATTR7	000027	DMKGRF	DMKGRT	DMKGRW	DMKRGA	DMKRGB							
AUTGO	000013	DMKCPI	DMKENT	DMKMCC	DMKMCD	DMKMID	DMKMNI						
AUTOSPL	000005	DMKMCD	DMKMIA	DMKMNI									
AVMREAL	000036	DMKBLD	DMKCFG	DMKCFP	DMKCFS	DMKCPV	DMKLOG	DMKMCH	DMKMCT	DMKPGS	DMKPTR	DMKRPA	DMKSCH
		DMKVIO	DMKVSI										
BALRSAVE	000097	DMKAPI	DMKCCW	DMKCFM	DMKCNS	DMKCPI	DMKCPV	DMKCSO	DMKCVT	DMKDIA	DMKFRE	DMKGRT	DMKLOC
		DMKPGT	DMKPTR	DMKQCN	DMKRGA	DMKRGB	DMKRNH	DMKSCH	DMKSCN	DMKUNT	DMKVAT	DMKVCA	DMKVMA
		DMKVSI											
BALR0	000007	DMKCPI	DMKPGT	DMKPTR	DMKVMA								
BALR1	000028	DMKAPI	DMKCPI	DMKCVT	DMKDIA	DMKPGT	DMKSCN	DMKVDC	DMKVDS				
BALR11	000003	DMKQCN	DMKSCH										
BALR12	000002	DMKVAT											
BALR13	000002	DMKVAT											
BALR14	000011	DMKCPI	DMKLOC	DMKTMR	DMKVAT	DMKVCA	DMKVDC						
BALR15	000001	DMKVCA											
BALR2	000033	DMKCCW	DMKCPI	DMKCVT	DMKDMP	DMKGRT	DMKPTR	DMKSCN	DMKTMR	DMKVCA	DMKVMA		
BALR3	000010	DMKCCW	DMKCNS	DMKSCN	DMKVCA								
BALR6	000005	DMKCNS	DMKCPI	DMKVDC									
BALR8	000006	DMKCPI	DMKPGT	DMKSCN	DMKVSI								
BALR9	000004	DMKCNS	DMKVCA										
BLANKS	000174	DMKCDS	DMKCFC	DMKCFD	DMKCFM	DMKCFO	DMKCFS	DMKCNS	DMKCPE	DMKCPS	DMKCQG	DMKCQH	DMKCQP
		DMKCQR	DMKCQY	DMKCSB	DMKCSO	DMKCSP	DMKCSQ	DMKCSU	DMKCSV	DMKDAS	DMKDDR	DMKDEF	DMKDIA
		DMKDSB	DMKERM	DMKGRF	DMKHVC	DMKJRL	DMKLD00E	DMKLNK	DMKLOG	DMKLOH	DMKMCC	DMKMCD	DMKMCH
		DMKMIA	DMKMNI	DMKMSG	DMKNES	DMKNET	DMKNLD	DMKNLE	DMKQCN	DMKRGA	DMKRNH	DMKRSP	DMKSCN
		DMKSPL	DMKSSS	DMKTCS	DMKTHI	DMKTRC	DMKTRD	DMKUDR	DMKUDU	DMKUSO	DMKVCA	DMKVCN	DMKVDA
		DMKVDC	DMKVDD	DMKVDE	DMKVDR	DMKVDS							
BLK	000001	DMKWRM											
BLKMPX	000009	DMKAPI	DMKCLK	DMKCPI	DMKVSI								
BRAD	000006	DMKLD00E											
BRING	000364	DMKAPI	DMKATS	DMKCCH	DMKCCW	DMKCDB	DMKCDM	DMKCDS	DMKCFD	DMKCFG	DMKCFH	DMKCKS	DMKCNS
		DMKCPB	DMKCPI	DMKCPS	DMKCPU	DMKCPV	DMKCQH	DMKCSB	DMKCSO	DMKCST	DMKCSU	DMKDGD	DMKDRD
		DMKDSP	DMKERM	DMKGIO	DMKGRF	DMKGRT	DMKHVC	DMKHVD	DMKIOF	DMKIOG	DMKMCC	DMKMIA	DMKMON
		DMKNLD	DMKNLE	DMKPRG	DMKPRV	DMKPTR	DMKQCN	DMKRGA	DMKRGB	DMKRPA	DMKRSP	DMKSEP	DMKSNC
		DMKSPL	DMKSSS	DMKSVC	DMKTCS	DMKTMR	DMKTRC	DMKTRD	DMKTRK	DMKUDR	DMKUDU	DMKVAT	DMKVCH
		DMKVCN	DMKVDR	DMKVER	DMKVIO	DMKVMC	DMKVSI	DMKVSP	DMKVSQ	DMKWRM			
BSCAUSER	000004	DMKRGA	DMKRGB										
BSCBLOK	000005	DMKBSC	DMKRGA	DMKRGB									
BSCCNT	000014	DMKRGA											
BSCCOPY	000009	DMKRGA											
BSCECCW1	000004	DMKRGA											

LABEL	COUNT	REFERENCES											
BSCECCW2	000004	DMKRGA											
BSCENQ	000005	DMKRGA											
BSCETB	000005	DMKRGA											
BSCFLAG	000046	DMKRGA	DMKRGB										
BSCFLAG1	000022	DMKRGA	DMKRGB										
BSCFORCE	000003	DMKRGA											
BSCHALT	000003	DMKRGA	DMKRGB										
BSCIGN	000004	DMKRGA											
BSCINBID	000004	DMKRGA	DMKRGB										
BSCINDEX	000005	DMKRGA											
BSCLINE	000002	DMKRGB											
BSCLOG	000005	DMKRGA											
BSCOPIED	000004	DMKRGA											
BSCPA1	000003	DMKRGA											
BSCPCCW1	000011	DMKBSC	DMKRGA	DMKRGB									
BSCPCCW2	000008	DMKBSC	DMKRGA	DMKRGB									
BSCPCCW3	000003	DMKRGA											
BSCPCCW4	000006	DMKRGA	DMKRGB										
BSCRCVD	000008	DMKRGA	DMKRGB										
BSCREAD	000019	DMKBSC	DMKRGA	DMKRGB									
BSCREGEN	000003	DMKRGA											
BSCRESP	000052	DMKBSC	DMKRGA	DMKRGB									
BSCRPTR	000009	DMKRGA											
BSCRROBN	000006	DMKRGA	DMKRGB										
BSCRSTRT	000002	DMKRGA											
BSCRVI	000003	DMKRGA											
BSCSCAN	000007	DMKRGA	DMKRGB										
BSCSCCW1	000008	DMKRGA	DMKRGB										
BSCSCCW2	000005	DMKRGA	DMKRGB										
BSCSCCW3	000002	DMKRGA	DMKRGB										
BSCSEL	000012	DMKRGA	DMKRGB										
BSCSEND	000005	DMKRGA											
BSCSENSE	000015	DMKRGA											
BSCSIZE	000002	DMKRGA	DMKRGB										
BSCSIZE1	000005	DMKRGA	DMKRGB										
BSCSIZE2	000001	DMKRGB											
BSCSPTR	000011	DMKRGA	DMKRGB										
BSCTMRQ	000007	DMKRGA											
BSCTSTRQ	000004	DMKRGA											
BSCUCOPY	000006	DMKRGA											
BSCUECCW	000003	DMKRGA											
BUFAPL	000005	DMKGRF	DMKGRT	DMKGRW									
BUFCNT	000028	DMKALG	DMKCFM	DMKCFS	DMKCPI	DMKCST	DMKERM	DMKGRF	DMKRGA	DMKRSP	DMKUDU	DMKVCN	
BUFFER	000102	DMKALG	DMKCDM	DMKCFG	DMKCFM	DMKCFO	DMKCFS	DMKCPI	DMKCPS	DMKCSE	DMKCSO	DMKCSP	DMKCSQ
		DMKCST	DMKCSU	DMKCSV	DMKEMA	DMKERM	DMKGRF	DMKGRT	DMKLNK	DMKMSG	DMKNMT	DMKRGA	DMKRND
		DMKRSP	DMKSCN	DMKUDU	DMKVDC	DMKWRM							

LABEL	COUNT	REFERENCES											
BUFIN	000003	DMKCPI	DMKVCN										
BUFINLTH	000018	DMKCFM	DMKERM	DMKGRF	DMKLNK	DMKRGA	DMKRGB						
BUFNXT	000043	DMKALG	DMKCDM	DMKCFG	DMKCFM	DMKCFO	DMKCFS	DMKCPI	DMKCPS	DMKCSO	DMKCST	DMKCSU	DMKCSV
		DMKLNK	DMKMSG	DMKRSP	DMKSCN	DMKUDU	DMKVCN	DMKVDC					
BUFSIZE	000033	DMKALG	DMKCFM	DMKCPI	DMKERM	DMKGRF	DMKIMG	DMKLNK	DMKRGA	DMKRSP	DMKUDU		
BUSOUT	000006	DMKRNH	DMKRSE										
BUSY	000057	DMKCKP	DMKCNS	DMKCPI	DMKDDR	DMKDIR	DMKDMP	DMKFMT	DMKIOS	DMKPSA	DMKRNH	DMKSSP	DMKVCA
		DMKVIO	DMKVMI	DMKVSI									
CACTDEV	000002	DMKRNH											
CACTLIN	000002	DMKNET	DMKRNH										
CACTLTR	000002	DMKNES	DMKRNH										
CAW	000088	DMKCCH	DMKCKP	DMKCNS	DMKCPI	DMKDDR	DMKDIB	DMKDMP	DMKFMT	DMKIOS	DMKLD00E	DMKOPR	DMKSAV
		DMKSSP	DMKTRD	DMKVMI	DMKVSI								
CC	001134	DMKACO	DMKBSC	DMKCCW	DMKCFP	DMKCKP	DMKCNS	DMKCPI	DMKCSB	DMKDAS	DMKDDR	DMKDGD	DMKDIB
		DMKDIR	DMKDMP	DMKDSB	DMKFMT	DMKGRF	DMKGRT	DMKGRW	DMKIOS	DMKMCC	DMKMNI	DMKNLD	DMKNLE
		DMKOPR	DMKPAG	DMKRGA	DMKRGB	DMKRNH	DMKRSE	DMKRSP	DMKSAV	DMKSEP	DMKSPL	DMKSSP	DMKTAP
		DMKTCS	DMKTDK	DMKTRK	DMKUCB	DMKUCC	DMKUCS	DMKUDR	DMKVCA	DMKVCN	DMKVDE	DMKVMI	DMKVSI
		DMKVSP	DMKVSQ	DMKWRM									
CCC	000050	DMKBSC	DMKCCH	DMKCNS	DMKCPI	DMKDAS	DMKDSB	DMKDSP	DMKEIG	DMKGRF	DMKHVC	DMKIOE	DMKIOS
		DMKMSW	DMKRSE	DMKRSP	DMKSEV	DMKTAP	DMKUNT	DMKVIO	DMKVSI				
CCCPUID	000003	DMKCCH											
CCDESMD	000003	DMKDIA	DMKRNH										
CCDEVTYP	000001	DMKCCH											
CCHADDR	000002	DMKCCH											
CCHANID	000008	DMKCCH											
CCHCAV	000001	DMKCCH											
CCHCHNL	000012	DMKSEV	DMKSIX										
CCHCLOGL	000005	DMKCCH											
CCHCMDV	000010	DMKEIG	DMKSEV	DMKSIX									
CCHCNTB	000005	DMKSEV	DMKSIX										
CCHCPU	000003	DMKSEV	DMKSIX										
CCHCUA	000002	DMKCCH											
CCHDAV	000010	DMKEIG	DMKSEV	DMKSIX									
CCHDI	000003	DMKEIG	DMKSEV	DMKSIX									
CCHHIO	000002	DMKCCH											
CCHINTB	000001	DMKCCH											
CCHINTFC	000007	DMKSEV	DMKSIX										
CCHIOH	000001	DMKCCH											
CCHLOG45	000002	DMKCCH											
CCHLOG60	000001	DMKSIX											
CCHLOG70	000001	DMKSEV											
CCHLOG80	000002	DMKCCH	DMKEIG										
CCHRCV	000003	DMKCCH	DMKEIG										
CCHREC	000005	DMKCCH	DMKEIG	DMKSEV	DMKSIX								
CCHSIOB	000002	DMKCCH											
CCHSIZE	000002	DMKCCH											

LABEL	COUNT	REFERENCES											
CCHSIZE1	000002	DMKCCH											
CCHSNSB	000001	DMKCCH											
CCHSTG	000004	DMKSEV	DMKSIX										
CCHTIO	000002	DMKCCH											
CCHUSV	000005	DMKEIG	DMKSEV	DMKSIX									
CCPADDR	000001	DMKSNC											
CCPARM	000004	DMKNLD	DMKSNC										
CCPENTRY	000001	DMKNLD											
CCPMAXID	000001	DMKNLD											
CCPNAME	000003	DMKNLD	DMKSNC										
CCPPSIZE	000005	DMKNLD	DMKSNC										
CCPRESID	000002	DMKNLD											
CCPROGID	000003	DMKCCH											
CCPRSTAT	000001	DMKNLD											
CCPRSTEP	000001	DMKNLD											
CCPRSTYP	000001	DMKNLD											
CCPSIZE	000003	DMKNLD	DMKSNC										
CCPTNCP	000001	DMKNLD											
CCPTPEP	000001	DMKNLD											
CCPTYPE	000002	DMKNLD											
CCRECTYP	000002	DMKCCH											
CCW1	000012	DMKDAS	DMKLD00E	DMKUDR									
CCW2	000002	DMKDAS											
CCW3	000003	DMKDAS											
CD	000111	DMKCCW	DMKCNS	DMKDAS	DMKDDR	DMKDGD	DMKDIB	DMKDIR	DMKFMT	DMKGRF	DMKGRT	DMKGRW	DMKISM
		DMKOPR	DMKRGA	DMKRGB	DMKSSP	DMKTAP	DMKTRK	DMKUNT	DMKVCA	DMKVCN	DMKVMI	DMKVSI	DMKVSP
CDC	000048	DMKBSC	DMKCCH	DMKCNS	DMKDAS	DMKDSB	DMKDSP	DMKGRF	DMKHVC	DMKIOE	DMKIOF	DMKIOS	DMKMSW
		DMKNLD	DMKNLE	DMKRNH	DMKRSE	DMKRSP	DMKTAP	DMKUNT	DMKVIO	DMKVSI			
CDCTLIN	000001	DMKNET											
CDISPLY	000001	DMKNES											
CE	000089	DMKCKP	DMKCNS	DMKCPI	DMKDDR	DMKDIB	DMKDIR	DMKDMP	DMKFMT	DMKGRF	DMKHVC	DMKIOG	DMKIOS
		DMKNLD	DMKPAG	DMKRGA	DMKRSE	DMKRSP	DMKSAV	DMKSSP	DMKVCA	DMKVCN	DMKVIO	DMKVMI	DMKVSI
		DMKVSP											
CFSTOP	000009	DMKCPS	DMKMCD	DMKMIA	DMKMNI	DMKMON							
CHANID	000004	DMKIOG	DMKPRV										
CHANO	000001	DMKEXT											
CHBATTN	000013	DMKVCA											
CHBCENT	000003	DMKVCA											
CHBCNTL	000002	DMKVCA											
CHBEOFL	000014	DMKVCA											
CHBHIO	000015	DMKVCA											
CHBMNOP	000005	DMKVCA											
CHBM370	000020	DMKVCA	DMKVSI										
CHBRDBK	000007	DMKVCA											
CHBREAD	000008	DMKVCA											
CHBREST	000012	DMKVCA											

LABEL	COUNT	REFERENCES										
CHBSIZE	000003	DMKDIA	DMKVCA									
CHBWAIT	000014	DMKCFP	DMKVCA									
CHBWEOF	000002	DMKVCA										
CHBWRIT	000009	DMKVCA										
CHC	000014	DMKBSC	DMKCNS	DMKGRF	DMKHVC	DMKIOS	DMKRNH	DMKRSE	DMKTAP	DMKUNT		
CHGRDV	000004	DMKCSO										
CHGREGS	000002	DMKTMR										
CHGSFB	000023	DMKCKS	DMKCSQ	DMKCSU	DMKCSV	DMKDMP	DMKMIA	DMKRSP	DMKSPL	DMKVSP		
CHGSHQ	000008	DMKCSQ	DMKWRM									
CHNGMSG	000002	DMKCPI										
CHXBLOK	000013	DMKCFP	DMKCQG	DMKDIA	DMKVCA	DMKVSI						
CHXCMDB	000010	DMKVCA										
CHXCMDT	000014	DMKVCA										
CHXCNCT	000009	DMKCFP	DMKVCA									
CHXDATN	000005	DMKVCA										
CHXFLAG	000057	DMKCFP	DMKVCA	DMKVSI								
CHXIDAW	000004	DMKVCA										
CHXNCCW	000012	DMKVCA										
CHXOTHR	000009	DMKCQG	DMKDIA	DMKVCA								
CHXPKEY	000005	DMKVCA										
CHXRCNT	000010	DMKVCA										
CHXSTAT	000020	DMKVCA										
CHXYADD	000007	DMKCQG	DMKDIA	DMKVCA								
CHYBLOK	000005	DMKDIA	DMKVCA									
CHYCMDB	000001	DMKVCA										
CHYCMDT	000003	DMKVCA										
CHYCNCT	000004	DMKVCA										
CHYDATN	000006	DMKVCA										
CHYFLAG	000032	DMKVCA										
CHYIDAW	000001	DMKVCA										
CHYNCCW	000004	DMKVCA										
CHYOTHR	000001	DMKDIA										
CHYRCNT	000005	DMKVCA										
CHYSTAT	000003	DMKVCA										
CHYXADD	000004	DMKDIA	DMKVCA									
CKCMASK	000005	DMKAPI	DMKCLK	DMKCPI								
CKPBITS	000003	DMKRNH										
CKPBKSZ	000001	DMKRNH										
CKPBLOK	000004	DMKRNH	DMKWRM									
CKPNAME	000003	DMKRNH	DMKWRM									
CKPRMAX	000002	DMKRNH	DMKWRM									
CKPSIZE	000003	DMKRNH	DMKWRM									
CL	000007	DMKDAS	DMKDSB	DMKJRL	DMKVDE							
CLASDASD	000170	DMKACO	DMKCCW	DMKCFP	DMKCKP	DMKCPI	DMKCPS	DMKCPV	DMKCQG	DMKCQP	DMKCQR	DMKDDR	DMKDEF
		DMKDGD	DMKDIR	DMKDMP	DMKGIO	DMKIOC	DMKIOE	DMKIOF	DMKIOS	DMKLNK	DMKLOG	DMKMNI	DMKMSW
		DMKPRV	DMKSCN	DMKSSP	DMKSSS	DMKTRD	DMKVCH	DMKVDA	DMKVDC	DMKVDD	DMKVDE	DMKVDR	DMKVDS

LABEL	COUNT	REFERENCES											
		DMKVER	DMKVIO	DMKVMI	DMKVSI								
CLASGRAF	000074	DMKACO	DMKBLD	DMKCCW	DMKCFM	DMKCFP	DMKCFT	DMKCKP	DMKCPI	DMKCPS	DMKCPV	DMKCQG	DMKCQP
		DMKDEF	DMKDIA	DMKDIR	DMKGRF	DMKHVD	DMKIOF	DMKIOS	DMKOPR	DMKPSA	DMKQCN	DMKSSP	DMKTRD
		DMKUSO	DMKVCH	DMKVCN	DMKVDC	DMKVDS	DMKVIO	DMKVSI					
CLASSPEC	000075	DMKBLD	DMKCCW	DMKCFP	DMKCFT	DMKCKP	DMKCPB	DMKCPI	DMKCQG	DMKCQP	DMKCQR	DMKCQY	DMKDEF
		DMKDIA	DMKDIR	DMKHVD	DMKIOF	DMKIOS	DMKLOG	DMKLOH	DMKNES	DMKNET	DMKNLD	DMKNLE	DMKQCN
		DMKRNH	DMKSCN	DMKTRD	DMKUSO	DMKVCH	DMKVDA	DMKVDC	DMKVDR	DMKVDS	DMKVIO	DMKVMI	DMKVSI
		DMKWRM											
CLASTAPE	000088	DMKCCW	DMKCFO	DMKCFP	DMKCKP	DMKCPB	DMKCPI	DMKCPS	DMKCQG	DMKCQP	DMKCQR	DMKDDR	DMKDMP
		DMKGIO	DMKIOE	DMKIOF	DMKIOS	DMKMCC	DMKMNI	DMKSSP	DMKVCH	DMKVDC	DMKVDD	DMKVDE	DMKVDR
		DMKVDS	DMKVMI	DMKVSI									
CLASTERM	000148	DMKACO	DMKBLD	DMKCCW	DMKCFM	DMKCFP	DMKCFT	DMKCKP	DMKCNS	DMKCPB	DMKCPI	DMKCPS	DMKCPV
		DMKCQG	DMKCQP	DMKCQR	DMKCQY	DMKCSP	DMKCSQ	DMKCST	DMKDDR	DMKDEF	DMKDIA	DMKDIR	DMKHVD
		DMKIOC	DMKIOF	DMKIOS	DMKLOG	DMKLOH	DMKNES	DMKNET	DMKPSA	DMKQCN	DMKRGA	DMKSCN	DMKSSP
		DMKTRD	DMKUSO	DMKVCH	DMKVCN	DMKVDA	DMKVDC	DMKVDE	DMKVDR	DMKVDS	DMKVIO	DMKVSI	DMKWRM
CLASURI	000085	DMKCCW	DMKCFP	DMKCKP	DMKCPB	DMKCPS	DMKCQG	DMKCQP	DMKCSB	DMKCSO	DMKCSP	DMKCSQ	DMKCST
		DMKCSU	DMKCSV	DMKDEF	DMKDIR	DMKDRD	DMKHVD	DMKIOF	DMKIOS	DMKRSE	DMKRSP	DMKSCN	DMKSPL
		DMKSSP	DMKTRD	DMKVCH	DMKVDA	DMKVDC	DMKVDD	DMKVDR	DMKVDS	DMKVIO	DMKVMI	DMKVSI	DMKVSP
CLASURO	000081	DMKCCW	DMKCFO	DMKCFP	DMKCKP	DMKCPB	DMKCPI	DMKCPS	DMKCQG	DMKCQP	DMKCSB	DMKCSO	DMKCSP
		DMKCSQ	DMKCST	DMKDEF	DMKDIR	DMKDMP	DMKHVD	DMKIOE	DMKIOF	DMKIOS	DMKRSE	DMKRSP	DMKSCN
		DMKSSP	DMKTRD	DMKVCH	DMKVDA	DMKVDC	DMKVDD	DMKVDR	DMKVDS	DMKVIO	DMKVSI	DMKVSP	
CLCMD	000003	DMKMIA											
CLSUS	000010	DMKMIA	DMKMNI	DMKMON									
CMD	000002	DMKLD00E											
CMDREJ	000013	DMKCNS	DMKDIB	DMKIOS	DMKRNH	DMKRSE	DMKVCN	DMKVSP					
CNTLBTU	000005	DMKRNH											
CODE	000021	DMKCNS	DMKCPI	DMKDSP	DMKEXT	DMKFRE	DMKIOS	DMKLOK	DMKMCH	DMKPRG	DMKPSA	DMKRNH	DMKSCH
		DMKSVC	DMKVSI										
COMPFES	000009	DMKEIG	DMKSEV	DMKSIX									
COMPSEL	000015	DMKEIG	DMKSEV	DMKSIX									
COMPSYS	000006	DMKCCH	DMKEIG	DMKSEV									
CONACTV	000028	DMKCNS	DMKGRF	DMKRGA	DMKRNH								
CONADDR	000033	DMKCNS	DMKGRF	DMKMON	DMKQCN	DMKRGA	DMKRGB	DMKRNH					
CONCCW1	000085	DMKCNS	DMKDIR	DMKGRF	DMKRGA	DMKRGB	DMKRNH						
CONCCW2	000038	DMKCNS	DMKGRF	DMKRGA	DMKRGB	DMKRNH							
CONCCW3	000032	DMKCNS	DMKDIA	DMKIOE	DMKNES	DMKNET	DMKRGA	DMKRGB	DMKRNH				
CONCCW4	000031	DMKCNS	DMKGRF	DMKRGA	DMKRGB								
CONCNT	000073	DMKCNS	DMKGRF	DMKMON	DMKQCN	DMKRGA	DMKRGB	DMKRNH					
CONCNTL	000024	DMKCNS	DMKQCN	DMKRGA	DMKRGB	DMKRNH							
CONCOMND	000008	DMKCNS	DMKRNH										
CONDATA	000082	DMKCNS	DMKDIA	DMKGRF	DMKIOE	DMKNES	DMKQCN	DMKRGA	DMKRGB	DMKRNH			
CONDCNT	000029	DMKDIA	DMKIOE	DMKRGA	DMKRNH								
CONDEST	000004	DMKRNH											
CONDWC	000022	DMKGRF	DMKQCN	DMKRGB									
CONESCP	000021	DMKCNS	DMKGRF	DMKRGA	DMKRGB	DMKRNH							
CONEXTR	000001	DMKRNH											

LABEL	COUNT	REFERENCES											
CONFLAG	000007	DMKCNS	DMKRNH										
CONLABEL	000030	DMKRGA	DMKRGB										
CONOUTPT	000031	DMKCNS	DMKGRF	DMKQCN	DMKRGB	DMKRNH							
CONPARM	000116	DMKCNS	DMKDDR	DMKDIR	DMKFMT	DMKGRF	DMKQCN	DMKRGA	DMKRGB	DMKRNH	DMKSSP		
CONPNT	000091	DMKCNS	DMKGRF	DMKQCN	DMKRGA	DMKRGB	DMKRNH						
CONREAD	000001	DMKCFM											
CONRESP	000017	DMKCNS	DMKGRF	DMKQCN	DMKRGA	DMKRGB	DMKRNH						
CONRETN	000028	DMKCNS	DMKGRF	DMKQCN	DMKRGA	DMKRGB	DMKRNH						
CONRTAG	000003	DMKRNH											
CONRTRY	000012	DMKCNS	DMKRNH										
CONSOLE	000001	DMKDIR											
CONSPLT	000014	DMKCNS	DMKQCN	DMKRNH									
CONSRID	000014	DMKRNH											
CONSTAT	000127	DMKCNS	DMKGRF	DMKQCN	DMKRGA	DMKRGB	DMKRNH						
CONSYNC	000006	DMKCNS	DMKGRF	DMKQCN	DMKRGB	DMKRNH							
CONSYSR	000041	DMKDIA	DMKNES	DMKNET	DMKRNH								
CONTACT	000003	DMKNET	DMKRNH										
CONTASK	000120	DMKCNS	DMKGRF	DMKMON	DMKNES	DMKQCN	DMKRGA	DMKRGB	DMKRNH				
CONTCMD	000027	DMKRNH											
CONTSIZE	000037	DMKCNS	DMKGRF	DMKQCN	DMKRGA	DMKRGB	DMKRNH						
CONTSKSZ	000014	DMKCNS	DMKGRF	DMKQCN	DMKRGA	DMKRGB	DMKRNH						
CONUSER	000013	DMKCNS	DMKDIR	DMKQCN	DMKRGA	DMKRGB	DMKRNH						
CORBPNT	000013	DMKATS	DMKPTR										
CORCFLCK	000021	DMKATS	DMKBLD	DMKCPI	DMKCPV	DMKPGS	DMKPTR	DMKRPA	DMKVMA				
CORCP	000016	DMKACO	DMKCPI	DMKCPU	DMKDMP	DMKMCC	DMKMNI	DMKPTR					
CORDISA	000001	DMKMCH											
CORFLAG	000098	DMKACO	DMKATS	DMKBLD	DMKCCW	DMKCDS	DMKCFC	DMKCPI	DMKCPU	DMKCPV	DMKDGD	DMKDMP	DMKMCC
		DMKMCH	DMKMNI	DMKPGS	DMKPSA	DMKPTR	DMKRPA	DMKVMA					
CORFLUSH	000010	DMKATS	DMKCCW	DMKDGD	DMKPTR	DMKVMA							
CORFPNT	000053	DMKATS	DMKBLD	DMKCPI	DMKCPU	DMKCPV	DMKDMP	DMKFRE	DMKPGS	DMKPTR	DMKRPA	DMKUDR	
CORFREE	000009	DMKATS	DMKCPI	DMKPGS	DMKPTR	DMKVMA							
CORIOLCK	000016	DMKATS	DMKBLD	DMKCPI	DMKCPU	DMKMCH	DMKPGS	DMKPTR	DMKRPA	DMKVMA			
CORLCNT	000009	DMKBLD	DMKCPI	DMKPTR									
CORPGPNT	000039	DMKATS	DMKBLD	DMKCDS	DMKCPI	DMKCPU	DMKDGD	DMKFRE	DMKMCH	DMKPGS	DMKPTR	DMKRPA	DMKUDR
		DMKUNT	DMKVMA										
CORRSV	000023	DMKATS	DMKCFO	DMKCPU	DMKCPV	DMKPGS	DMKPTR	DMKRPA	DMKVMA				
CORSHARE	000023	DMKATS	DMKCCW	DMKCDS	DMKCPU	DMKCPV	DMKPGS	DMKPSA	DMKPTR				
CORSWPNT	000027	DMKATS	DMKBLD	DMKCCW	DMKCDS	DMKCPI	DMKCPU	DMKDGD	DMKMCH	DMKPGS	DMKPTR	DMKRPA	DMKUDU
		DMKVMA											
CORTABLE	000082	DMKACO	DMKATS	DMKBLD	DMKCCW	DMKCDS	DMKCFC	DMKCPI	DMKCPU	DMKCPV	DMKDGD	DMKDMP	DMKFRE
		DMKMCC	DMKMCH	DMKMNI	DMKPAG	DMKPGS	DMKPSA	DMKPTR	DMKRPA	DMKUDR	DMKUDU	DMKUNT	DMKVMA
CORVM	000010	DMKATS	DMKCPU	DMKCPV	DMKPTR	DMKVMA							
COUNT	000022	DMKCSU	DMKCSV	DMKERM	DMKFMT	DMKFRE	DMKIOS	DMKTRA					
CPABEND	000006	DMKDMP	DMKPRG	DMKPSA	DMKSVC								
CPAPRPND	000005	DMKDSP	DMKEXT	DMKMCT									
CPASTAVL	000007	DMKCFO	DMKCFS	DMKCPI	DMKLOG								

LABEL	COUNT	REFERENCES											
CPASTON	000007	DMKCFO	DMKCPI	DMKCQY	DMKLOG								
CPCREG0	000037	DMKAPI	DMKCKP	DMKCLK	DMKCPI	DMKCPU	DMKDSP	DMKEXT	DMKIOS	DMKLOK	DMKMCH	DMKPRG	DMKPRV
		DMKPSA	DMKSVC	DMKTMR	DMKTRC	DMKTRD	DMKVAT						
CPCREG6	000007	DMKCFO	DMKCPI	DMKDSP									
CPCREG8	000034	DMKCPS	DMKIOS	DMKMCC	DMKMCD	DMKMCH	DMKMIA	DMKMNI	DMKMON	DMKPRG	DMKPSA	DMKSVC	
CPEX	000008	DMKDSP	DMKVCA										
CPEXADD	000099	DMKACO	DMKALG	DMKCCW	DMKCDS	DMKCFM	DMKCFC	DMKCFP	DMKCNS	DMKCPS	DMKCPU	DMKCPV	DMKDGD
		DMKDIA	DMKDSB	DMKDSP	DMKEXT	DMKFRE	DMKGIC	DMKGRF	DMKIOE	DMKIOS	DMKLOC	DMKLOK	DMKMCC
		DMKMCD	DMKMCH	DMKMCT	DMKMIA	DMKMID	DMKMNI	DMKMON	DMKPAG	DMKPGS	DMKPGT	DMKPRG	DMKPRV
		DMKPSA	DMKPTR	DMKQCN	DMKRGA	DMKRGB	DMKRNH	DMKRPA	DMKRSP	DMKSPL	DMKSSS	DMKSVC	DMKTMR
		DMKTRD	DMKUSO	DMKVAT	DMKVCA	DMKVMA	DMKVMC	DMKVSI	DMKVSP				
CPEXBLOK	000186	DMKACO	DMKALG	DMKCCW	DMKCDS	DMKCFM	DMKCFC	DMKCFP	DMKCNS	DMKCPB	DMKCPS	DMKCPU	DMKCPV
		DMKDGD	DMKDIA	DMKDSB	DMKDSP	DMKEXT	DMKFRE	DMKGIO	DMKGRF	DMKIOE	DMKIOF	DMKIOS	DMKLNK
		DMKLOC	DMKLOK	DMKMCC	DMKMCD	DMKMCH	DMKMCT	DMKMIA	DMKMID	DMKMNI	DMKMON	DMKPAG	DMKPGS
		DMKPGT	DMKPRG	DMKPRV	DMKPSA	DMKPTR	DMKQCN	DMKRGA	DMKRGB	DMKRNH	DMKRPA	DMKRSP	DMKSPL
		DMKSSS	DMKSTK	DMKSVC	DMKTAP	DMKTMR	DMKTRD	DMKUNT	DMKUSO	DMKVAT	DMKVCA	DMKVDA	DMKVDC
		DMKVDE	DMKVMA	DMKVMC	DMKVSI	DMKVSP							
CPEXBPNT	000004	DMKDSP	DMKPAG										
CPEXDEFR	000005	DMKCPU	DMKDSP	DMKSTK									
CPEXFPNT	000048	DMKCCW	DMKCDS	DMKCFP	DMKDGD	DMKDSP	DMKGIC	DMKIOE	DMKIOF	DMKLOC	DMKPAG	DMKPTR	DMKRPA
		DMKUNT	DMKVCA	DMKVSP									
CPEXLPSW	000002	DMKDSP	DMKSVC										
CPEXMISC	000030	DMKCCW	DMKCFP	DMKCPB	DMKDGD	DMKGIO	DMKPAG	DMKPTR	DMKSSS	DMKSTK	DMKVSI		
CPEXPRIO	000006	DMKDSP	DMKSTK										
CPEXPROC	000021	DMKCPU	DMKDSP	DMKFRE	DMKIOS	DMKLOC	DMKLOK	DMKPRG	DMKSTK				
CPEXREGS	000044	DMKCFM	DMKCNS	DMKCPS	DMKCPU	DMKCPV	DMKDGD	DMKDSB	DMKDSP	DMKEXT	DMKIOE	DMKIOF	DMKLOC
		DMKLOK	DMKMCD	DMKMCH	DMKMCT	DMKPRG	DMKPRV	DMKPSA	DMKQCN	DMKSPL	DMKSSS	DMKSVC	DMKTAP
		DMKTMR	DMKVAT	DMKVMA	DMKVSI								
CPEXR0	000052	DMKALG	DMKCCW	DMKCDS	DMKCFP	DMKCPS	DMKDGD	DMKFRE	DMKGIO	DMKGRF	DMKIOS	DMKMCC	DMKMIA
		DMKMID	DMKMNI	DMKMON	DMKPAG	DMKPGS	DMKPTR	DMKRGA	DMKRPA	DMKSSS	DMKTAP	DMKUSO	DMKVCA
		DMKVDC	DMKVDE										
CPEXR1	000007	DMKCPU	DMKIOS	DMKLNK	DMKLOK	DMKSSS	DMKVDA	DMKVSP					
CPEXR10	000005	DMKCNS	DMKDSP	DMKMIA									
CPEXR11	000031	DMKCFP	DMKCPU	DMKDSP	DMKEXT	DMKLOK	DMKMCC	DMKMIA	DMKMID	DMKPAG	DMKPGT	DMKPSA	DMKPTR
		DMKSSS	DMKSTK	DMKUSO	DMKVMC	DMKVSP							
CPEXR12	000017	DMKALG	DMKCPS	DMKCPU	DMKCPV	DMKMCC	DMKMIA	DMKMID	DMKMNI	DMKQCN	DMKSSS	DMKSVC	DMKUSO
		DMKVCA	DMKVMC										
CPEXR13	000009	DMKIOS	DMKPTR	DMKVDE									
CPEXR14	000002	DMKCDS	DMKLOK										
CPEXR15	000002	DMKCDS	DMKDSP										
CPEXR2	000007	DMKCFO	DMKPTR	DMKQCN	DMKUSO								
CPEXR3	000001	DMKSSS											
CPEXR5	000006	DMKCDS	DMKPAG	DMKVDE									
CPEXR6	000005	DMKIOF											
CPEXR7	000003	DMKCDS	DMKPAG	DMKPTR									
CPEXR8	000002	DMKDSP	DMKVSP										

LABEL	COUNT	REFERENCES											
CPEXR9	000002	DMKPTR											
CPEXSIZE	000127	DMKACO	DMKALG	DMKAPI	DMKBLD	DMKCCW	DMKCDS	DMKCFM	DMKCFO	DMKCFP	DMKCNS	DMKCPI	DMKCPS
		DMKCPU	DMKCPV	DMKDGD	DMKDIA	DMKDSB	DMKDSP	DMKEXT	DMKFRE	DMKGIO	DMKGRF	DMKIOE	DMKIOF
		DMKIOG	DMKIOS	DMKLOC	DMKLOG	DMKLOK	DMKMCC	DMKMCD	DMKMCH	DMKMCT	DMKMIA	DMKMID	DMKMNI
		DMKMON	DMKPGS	DMKPGT	DMKPSA	DMKPTR	DMKQCN	DMKRGA	DMKRGB	DMKRNH	DMKRPA	DMKRSP	DMKSPL
		DMKSSS	DMKSVC	DMKTAP	DMKTRD	DMKUSO	DMKVCA	DMKVDC	DMKVDE	DMKVMA	DMKVMC	DMKVSI	DMKVSP
CPEXTYPE	000012	DMKCPU	DMKDSP	DMKSTK	DMKSVC								
CPFRELK	000007	DMKDSP	DMKFRE										
CPFRESW	000003	DMKDSP	DMKFRE										
CPID	000030	DMKCKP	DMKCNS	DMKCPI	DMKCPS	DMKCPU	DMKCVT	DMKDMP	DMKGRF	DMKMCH	DMKMCT	DMKPGT	
CPINITD	000005	DMKAPI	DMKCFM	DMKCPI	DMKIOS								
CPLOKFL	000006	DMKEXT	DMKLOK										
CPMCHLK	000002	DMKMCH											
CPMCHSE	000005	DMKDSP	DMKMCH										
CPMICAVL	000011	DMKAPI	DMKCFO	DMKCFS	DMKCPI	DMKLOG							
CPMICON	000014	DMKAPI	DMKCFO	DMKCFS	DMKCPI	DMKCQR	DMKCQY	DMKDSP	DMKPRV				
CPPTLBR	000022	DMKATS	DMKBLD	DMKCPU	DMKDSP	DMKMCH	DMKPGS	DMKPRV	DMKPTR	DMKRPA	DMKVAT	DMKVMA	
CPRUN	000020	DMKCFP	DMKDSP	DMKEXT	DMKIOS	DMKPSA	DMKSCH	DMKSVC	DMKTMR				
CPSHRLK	000007	DMKCCW	DMKDGD	DMKDSP									
CPSTATUS	000051	DMKAPI	DMKCFP	DMKCLK	DMKCPI	DMKDSP	DMKEXT	DMKIOS	DMKMCH	DMKMCT	DMKPRG	DMKPSA	DMKSCH
		DMKSVC	DMKTMR										
CPSTAT2	000035	DMKAPI	DMKCCW	DMKCFO	DMKCFS	DMKCPI	DMKCQR	DMKCQY	DMKDGD	DMKDSP	DMKLOG	DMKPRV	
CPSTAT3	000008	DMKDSP											
CPSUPER	000011	DMKCPI	DMKDSP	DMKIOS	DMKMCT	DMKPRG	DMKPSA	DMKSCH	DMKSVC				
CPSYSLK	000004	DMKDSP	DMKEXT	DMKLOK									
CPTERMLK	000003	DMKMCT											
CPTIDLE	000004	DMKDSP											
CPTIONT	000002	DMKDSP											
CPTMASK	000004	DMKAPI	DMKCLK										
CPTPAGE	000003	DMKDSP											
CPUID	000035	DMKAPI	DMKCCH	DMKCPI	DMKCQY	DMKDDR	DMKDIR	DMKHVC	DMKIOF	DMKIOG	DMKLOG	DMKMCH	DMKMNI
		DMKOPR	DMKPRV	DMKSSP	DMKVER								
CPULOG	000003	DMKCPI	DMKDMP										
CPUMCELL	000004	DMKCPU	DMKHVD	DMKIOG	DMKPRV								
CPUMODEL	000006	DMKCPI	DMKIOG	DMKMCC									
CPUSER	000001	DMKPRV											
CPUVERSN	000010	DMKHVD	DMKIOF	DMKIOG	DMKMCH	DMKOPR	DMKPRV	DMKSSP					
CPWAIT	000023	DMKAPI	DMKCLK	DMKCPI	DMKDSP	DMKEXT	DMKIOS	DMKMCH	DMKMCT	DMKPSA	DMKSCH	DMKVCA	
CRBIT	000001	DMKMCH											
CRESCND	000001	DMKRNH											
CRESDQ	000001	DMKDIA											
CRESERL	000002	DMKRNH											
CRESIMD	000005	DMKDIA	DMKNET	DMKPSA	DMKRNH								
CSETDSM	000002	DMKDIA	DMKRNH										
CSW	000288	DMKCCH	DMKCKP	DMKCNS	DMKCPI	DMKDDR	DMKDGD	DMKDIR	DMKDMP	DMKDSP	DMKEIG	DMKFMT	DMKGIO
		DMKIOS	DMKLD00E	DMKOPR	DMKPSA	DMKSAV	DMKSEV	DMKSIX	DMKSSP	DMKTRA	DMKTRD	DMKVCN	DMKVIO

LABEL	COUNT	REFERENCES											
		DMKVMI	DMKVSI	DMKVSP									
CSWLMEP	000002	DMKDIA	DMKNES										
CSWLNCP	000002	DMKDIA	DMKNES										
CTL	000001	DMKLD00E											
CTRMLTR	000003	DMKDIA	DMKNES	DMKRNH									
CUE	000047	DMKCFP	DMKCKP	DMKCPI	DMKDDR	DMKDIR	DMKDMP	DMKDSP	DMKFMT	DMKIOS	DMKMON	DMKNLD	DMKNLE
		DMKPAG	DMKPSA	DMKRSE	DMKSSP	DMKTAP	DMKVIO	DMKVSI					
CURRSAVE	000004	DMKIMG											
CVTEXIT	000001	DMKNEM											
C0	000105	DMKAPI	DMKCKP	DMKCLK	DMKCPI	DMKCPU	DMKDMP	DMKDSP	DMKEXT	DMKIOS	DMKLOK	DMKMCH	DMKPRG
		DMKPRV	DMKPSA	DMKSVC	DMKTMR	DMKTRC	DMKTRD						
C1	000417	DMKAPI	DMKATS	DMKBLD	DMKCCH	DMKCCW	DMKCDE	DMKCDM	DMKCDS	DMKCFD	DMKCFG	DMKCFH	DMKCFS
		DMKCKS	DMKCNS	DMKCPB	DMKCPI	DMKCPS	DMKCPU	DMKCPV	DMKCSE	DMKCSO	DMKDGD	DMKDRD	DMKDSP
		DMKERM	DMKEXT	DMKGIO	DMKGRF	DMKGRT	DMKHVC	DMKHVD	DMKIOG	DMKIOS	DMKISM	DMKMCC	DMKMCH
		DMKMIA	DMKMNI	DMKMON	DMKNLD	DMKNLE	DMKPGS	DMKPRG	DMKPRV	DMKPTR	DMKQCN	DMKRGA	DMKRGB
		DMKRSP	DMKSCH	DMKSEP	DMKSNC	DMKSPL	DMKSSS	DMKSVC	DMKTCS	DMKTMR	DMKTRA	DMKTRC	DMKTRD
		DMKTRK	DMKUDR	DMKUDU	DMKVAT	DMKVCH	DMKVCN	DMKVDR	DMKVER	DMKVIO	DMKVMC	DMKVSI	DMKVSP
		DMKVSQ	DMKWRM										
C11	000002	DMKDSP											
C13	000003	DMKDSP	DMKMCH										
C14	000012	DMKAPI	DMKCFH	DMKCPI	DMKDMP	DMKPRV							
C15	000004	DMKCFH	DMKDMP	DMKPRV									
C2	000046	DMKAPI	DMKCFH	DMKCKP	DMKCPI	DMKDMP	DMKEXT	DMKFRE	DMKIOS	DMKLD00E			
C3	000002	DMKCKP	DMKMCH										
C4	000002	DMKDSP											
C5	000001	DMKDSP											
C6	000018	DMKAPI	DMKCFO	DMKCPI	DMKDSP	DMKPRV							
C7	000004	DMKCCH	DMKDSP	DMKMCH									
C8	000034	DMKCPS	DMKEXT	DMKIOS	DMKMCC	DMKMCD	DMKMIA	DMKMNI	DMKMON	DMKPRG	DMKPSA	DMKSVC	
C9	000002	DMKDSP											
DAMAGRPT	000002	DMKCPI	DMKDMP										
DASDCL	000006	DMKMCC	DMKMNI	DMKMON									
DATACHK	000008	DMKCNS	DMKRSE	DMKUNT	DMKVSP								
DATAEND	000001	DMKVCA											
DATE	000034	DMKACO	DMKCKP	DMKCPI	DMKCVT	DMKDDR	DMKMID						
DDRCUA1	000002	DMKVER											
DDRCUA2	000002	DMKVER											
DDRKEYN	000001	DMKVER											
DDRREC	000001	DMKVER											
DDRSIZE	000001	DMKVER											
DE	000119	DMKACO	DMKCFP	DMKCKP	DMKCNS	DMKCPB	DMKCPI	DMKCPS	DMKCSO	DMKCSP	DMKCSU	DMKCSV	DMKDDR
		DMKDIA	DMKDIB	DMKDIR	DMKDMP	DMKDSB	DMKFMT	DMKGRF	DMKHVC	DMKIOG	DMKIOS	DMKLD00E	DMKMON
		DMKNLD	DMKNLE	DMKPAG	DMKRGA	DMKRNH	DMKRSE	DMKRSP	DMKSAV	DMKSPL	DMKSSP	DMKTAP	DMKUNT
		DMKVCA	DMKVCN	DMKVDD	DMKVIO	DMKVMI	DMKVSI	DMKVSP					
DECAREA	000001	DMKIMG											
DECDCBAD	000001	DMKIMG											

LABEL	COUNT	REFERENCES											
DEFER	000358	DMKAPI	DMKATS	DMKBLD	DMKCCH	DMKCCW	DMKCDB	DMKCDM	DMKCIS	DMKCFD	DMKCFG	DMKCFH	DMKCKS
		DMKCNS	DMKCPB	DMKCPI	DMKCPS	DMKCPU	DMKCPV	DMKCSB	DMKCSO	DMKDGD	DMKDRD	DMKDSP	DMKERM
		DMKGIO	DMKGRF	DMKGRT	DMKHVC	DMKHVD	DMKIOF	DMKIOG	DMKMCC	DMKMIA	DMKMON	DMKNLD	DMKNLE
		DMKPGS	DMKPRG	DMKPRV	DMKPTR	DMKQCN	DMKRGA	DMKRGB	DMKRPA	DMKRSP	DMKSEP	DMKSNC	DMKSPL
		DMKSSS	DMKSVC	DMKTCS	DMKTMR	DMKTRC	DMKTRD	DMKTRK	DMKUIR	DMKUDU	DMKVAT	DMKVCH	DMKVCN
		DMKVDR	DMKVER	DMKVIO	DMKVMC	DMKVSI	DMKVSP	DMKVSQ	DMKWRM				
DEFINTVL	000002	DMKMCD	DMKMNI										
DELPAGES	000013	DMKBLD	DMKCFP	DMKDEF	DMKUSO								
DELSEGS	000008	DMKBLD	DMKDEF	DMKUSO									
DELSFB	000007	DMKCKS	DMKCSQ	DMKSPL									
DEMOUNT	000002	DMKSSS											
DEVADDR	000037	DMKCSO	DMKVDA	DMKVDC	DMKVDD								
DEVCARD	000002	DMKACO	DMKCKP										
DEVCCH	000005	DMKCCH											
DEVCODE	000001	DMKIOC											
DEVICE	000009	DMKCPB	DMKCPS	DMKEMA	DMKLD00E	DMKNET							
DEVTYPE	000004	DMKCSO											
DFRET	000034	DMKACO	DMKCPS	DMKCQR	DMKCQY	DMKDAS	DMKDIA	DMKDSB	DMKERM	DMKJRL	DMKQCN	DMKRNH	DMKSVC
		DMKTHI	DMKVDD										
DIRPTR	000013	DMKDIR											
DISCEOC	000002	DMKRNH											
DISCNCT	000001	DMKRNH											
DISPATCH	000011	DMKDSP											
DISPMSG	000001	DMKACO											
DMKACO	000001	DMKSYM											
DMKACODV	000008	DMKCPV	DMKDIA	DMKVDR									
DMKACOFF	000006	DMKCPV	DMKUSO										
DMKACON	000002	DMKLOG											
DMKACOPU	000002	DMKRSP											
DMKACOQU	000008	DMKHVD	DMKJRL										
DMKACOTM	000006	DMKCPV	DMKCQY	DMKUSO									
DMKALG	000001	DMKSYM											
DMKALGON	000004	DMKCFC	DMKCPI										
DMKAPI	000001	DMKSYM											
DMKAPIPR	000004	DMKCPI	DMKCPU										
DMKATS	000001	DMKSYM											
DMKATSCF	000010	DMKCDS	DMKCFD	DMKTRC	DMKVAT								
DMKBLD	000001	DMKSYM											
DMKBLDEC	000004	DMKCFS	DMKLOG										
DMKBLDRL	000010	DMKCFP	DMKDEF	DMKPGS	DMKUSO								
DMKBLDRT	000014	DMKCFG	DMKCFP	DMKCPI	DMKDEF	DMKLOG	DMKPGS	DMKPTR					
DMKBLDVM	000012	DMKALG	DMKCNS	DMKDIA	DMKGRF	DMKRGA	DMKRNH						
DMKBOX	000001	DMKSYM											
DMKBOXBX	000003	DMKGRT	DMKRGB	DMKSEP									
DMKBOXHR	000001	DMKVSQ											
DMKBSCER	000002	DMKIOS	DMKSYM										

LABEL	COUNT	REFERENCES				
DMKCCHCF	000001	DMKIOG				
DMKCCHIS	000005	DMKIOS	DMKSYM			
DMKCCHMX	000001	DMKIOG				
DMKCCHNT	000005	DMKIOS	DMKSYM			
DMKCCHRF	000006	DMKDSP	DMKVIO	DMKVSI		
DMKCCHRT	000003	DMKIOE	DMKSYM			
DMKCCHSZ	000001	DMKIOG				
DMKCCH60	000002	DMKIOG	DMKSYM			
DMKCCWB1	000001	DMKCPI				
DMKCCWB2	000001	DMKCPI				
DMKCCWB3	000001	DMKCPI				
DMKCCWB4	000001	DMKCPI				
DMKCCWB5	000001	DMKCPI				
DMKCCWB6	000001	DMKCPI				
DMKCCWB7	000001	DMKCPI				
DMKCCWB8	000001	DMKCPI				
DMKCCWGN	000001	DMKCPI				
DMKCCWL1	000001	DMKCPI				
DMKCCWL2	000001	DMKCPI				
DMKCCWL3	000001	DMKCPI				
DMKCCWL4	000001	DMKCPI				
DMKCCWL5	000001	DMKCPI				
DMKCCWSB	000005	DMKSYM	DMKTRD	DMKTRK		
DMKCCWTC	000002	DMKHVC	DMKSYM			
DMKCCWTR	000007	DMKGIO	DMKHVC	DMKSYM	DMKVSI	
DMKCCW0	000001	DMKCPI				
DMKCCW1	000001	DMKCPI				
DMKCDB	000001	DMKSYM				
DMKCDBDC	000003	DMKCFC	DMKSYM			
DMKCDBDI	000002	DMKCFC				
DMKCDM	000001	DMKSYM				
DMKCDMDM	000002	DMKCFC				
DMKCDMDU	000002	DMKCFC				
DMKCDS	000001	DMKSYM				
DMKCDSCP	000002	DMKCFC				
DMKCDSTO	000002	DMKCFC				
DMKCFC	000001	DMKSYM				
DMKCFCMD	000002	DMKCFM				
DMKCFCSC	000010	DMKCPS	DMKCQG	DMKCQP	DMKMCD	DMKVDC
DMKCFD	000001	DMKSYM				
DMKCFDAD	000003	DMKCFC	DMKSYM			
DMKCFDLO	000002	DMKCFC				
DMKCFG	000001	DMKSYM				
DMKCFGCL	000002	DMKHVC				
DMKCFGII	000002	DMKLOG				
DMKCFGIP	000002	DMKCFC				

LABEL	COUNT	REFERENCES											
DMKCFH	000001	DMKSYM											
DMKCFHSV	000002	DMKCFC											
DMKCFMAT	000019	DMKCFC	DMKCNS	DMKGRF	DMKRGA	DMKRNH	DMKSYM	DMKVCN					
DMKCFMBK	000041	DMKCNS	DMKCPS	DMKDSP	DMKGRF	DMKHVC	DMKMCH	DMKMCT	DMKPRG	DMKRGA	DMKRNH	DMKSVC	DMKSYM
		DMKTRC	DMKTRD	DMKVCN	DMKVMA								
DMKCFMEN	000011	DMKCPI	DMKGRF	DMKHVC	DMKRGA	DMKSYM	DMKVCN						
DMKCFMWU	000001	DMKCFC											
DMKCFOEX	000002	DMKCFC	DMKSYM										
DMKCFP	000001	DMKSYM											
DMKCFPRD	000014	DMKCPB	DMKCPS	DMKDEF	DMKDIA	DMKLNK	DMKSYM	DMKVDR					
DMKCFPRR	000020	DMKCFG	DMKCFS	DMKCPB	DMKDEF	DMKMCH	DMKMCT	DMKUSO					
DMKCFSET	000002	DMKCFC	DMKSYM										
DMKCFT	000001	DMKSYM											
DMKCFTRM	000002	DMKCFC											
DMKCKP	000007	DMKCPI	DMKPGT	DMKSAV									
DMKCKPLD	000001	DMKSAV											
DMKCKPRS	000001	DMKSAV											
DMKCKPST	000001	DMKSAV											
DMKCKPT	000002	DMKSAV	DMKSYM										
DMKCKS	000001	DMKSYM											
DMKCKSIN	000002	DMKWRM											
DMKCKSPL	000044	DMKCSO	DMKCSQ	DMKCSU	DMKCSV	DMKMIA	DMKNLE	DMKRSP	DMKSPL	DMKVSP	DMKWRM		
DMKCKSWM	000002	DMKWRM											
DMKCLK	000001	DMKSYM											
DMKCLKAP	000001	DMKEXT											
DMKCLKCC	000001	DMKEXT											
DMKCLKCK	000004	DMKCPI	DMKCPU										
DMKCLKSC	000002	DMKEXT											
DMKCNSED	000011	DMKGRF	DMKRGA	DMKRNH	DMKSYM								
DMKCNSEN	000004	DMKCPI	DMKCPV	DMKSYM									
DMKCNSIC	000002	DMKQCN	DMKSYM										
DMKCNSIN	000002	DMKIOS	DMKSYM										
DMKCPB	000001	DMKSYM											
DMKCPBEX	000002	DMKCFC											
DMKCPBNR	000002	DMKCFC											
DMKCPBRS	000002	DMKCFC											
DMKCPBRW	000002	DMKCFC											
DMKCPBRY	000002	DMKCFC											
DMKCPBSR	000002	DMKCFC											
DMKCPE	000003	DMKFRE	DMKLD00E	DMKPSA									
DMKCPEID	000005	DMKCPI	DMKHVD	DMKMNI	DMKSEP	DMKSYM							
DMKCPEML	000003	DMKCPI	DMKSYM										
DMKCPEND	000007	DMKCPI	DMKMNI	DMKSYM									
DMKCPEPP	000002	DMKHVD											
DMKCPI	000001	DMKSYM											
DMKCPICD	000001	DMKSAV											

LABEL	COUNT	REFERENCES			
DMKCPIEM	000004	DMKCNS	DMKGRF		
DMKCPINT	000002	DMKSAV	DMKSSP		
DMKCPS	000001	DMKSYM			
DMKCPSH	000003	DMKCFC	DMKSYM		
DMKCPSRY	000002	DMKCFC			
DMKCPSSH	000002	DMKCFC			
DMKCPU	000001	DMKSYM			
DMKCPUUP	000003	DMKMCT	DMKSYM		
DMKCPUVY	000003	DMKCPS	DMKSYM		
DMKCPV	000001	DMKSYM			
DMKCPVAA	000002	DMKHVD			
DMKCPVAC	000004	DMKCFC	DMKUDU		
DMKCPVAE	000004	DMKCPI	DMKRNH		
DMKCPVDS	000002	DMKCFC			
DMKCPVEN	000002	DMKCFC			
DMKCPVLK	000002	DMKCFC			
DMKCPVUL	000002	DMKCFC			
DMKCQG	000001	DMKSYM			
DMKCQGEN	000001	DMKCFC			
DMKCQHPR	000002	DMKCQG			
DMKCQHPU	000002	DMKCQG			
DMKCQHRD	000002	DMKCQG			
DMKCQP	000001	DMKSYM			
DMKCQPRV	000001	DMKCFC			
DMKCQR	000001	DMKSYM			
DMKCQREY	000001	DMKCFC			
DMKCQRFI	000004	DMKCPI	DMKLOH		
DMKCQRWS	000001	DMKCPU			
DMKCQY	000001	DMKSYM			
DMKCQYEY	000001	DMKCFC			
DMKCSB	000001	DMKSYM			
DMKCSBLD	000002	DMKCFC			
DMKCSBVL	000002	DMKCFC			
DMKCSO	000001	DMKSYM			
DMKCSOBS	000002	DMKCFC			
DMKCSODR	000002	DMKCFC			
DMKCSOFL	000002	DMKCFC			
DMKCSORP	000002	DMKCFC			
DMKCSOSD	000008	DMKCPI	DMKCSQ	DMKCSU	DMKRSP
DMKCSOSP	000002	DMKCFC			
DMKCSOST	000002	DMKCFC			
DMKCSP	000001	DMKSYM			
DMKCSPSP	000002	DMKCFC			
DMKCSQ	000001	DMKSYM			
DMKCSQCL	000002	DMKCFC			
DMKCSQFR	000002	DMKCFC			

LABEL	COUNT	REFERENCES											
DMKCSQHL	000002	DMKCFC											
DMKCST	000001	DMKSYM											
DMKCSTAG	000002	DMKCFC											
DMKCSU	000001	DMKSYM											
DMKCSUCH	000002	DMKCFC											
DMKCSV	000001	DMKSYM											
DMKCSVOR	000002	DMKCFC											
DMKCSVPU	000002	DMKCFC											
DMKCSVTR	000003	DMKCFC	DMKSYM										
DMKCVTAB	000050	DMKACO	DMKATS	DMKBLD	DMKCFC	DMKCQR	DMKDIA	DMKENT	DMKEXT	DMKGRF	DMKIOE	DMKIOG	DMKLOH
		DMKLOK	DMKMCC	DMKMNI	DMKMON	DMKPAG	DMKPGS	DMKPTR	DMKQCN	DMKRGA	DMKSCH	DMKTMR	DMKVDS
DMKCVTBD	000190	DMKCDB	DMKCDM	DMKCKS	DMKCPI	DMKCPS	DMKCQG	DMKCQH	DMKCQP	DMKCQR	DMKCQY	DMKCSU	DMKCSV
		DMKDEF	DMKDIA	DMKERM	DMKGRF	DMKJRL	DMKLNK	DMKLOH	DMKMIA	DMKMNI	DMKQCN	DMKRGA	DMKRSE
		DMKRSP	DMKSEP	DMKSPL	DMKTCS	DMKTHI	DMKUSC	DMKVDA	DMKVDD	DMKWRM			
DMKCVTBH	000550	DMKACO	DMKBLD	DMKCCH	DMKCDB	DMKCDM	DMKCDS	DMKCFD	DMKCFG	DMKCFH	DMKCFO	DMKCFS	DMKCFT
		DMKCNS	DMKCPB	DMKCPI	DMKCPS	DMKCPU	DMKCPV	DMKCQG	DMKCQP	DMKCQR	DMKCQY	DMKCSO	DMKCST
		DMKDAS	DMKDEF	DMKDIA	DMKDSB	DMKDSP	DMKJRL	DMKLNK	DMKLOG	DMKLOH	DMKMCD	DMKMSW	DMKNES
		DMKNET	DMKNLD	DMKNLE	DMKPAG	DMKQCN	DMKRGA	DMKRNH	DMKRSE	DMKRSP	DMKSAV	DMKSEP	DMKSSP
		DMKSSS	DMKSVC	DMKTCS	DMKTHI	DMKTRC	DMKTRD	DMKUSO	DMKVCA	DMKVCH	DMKVDA	DMKVDD	DMKVDR
		DMKVDS	DMKVER	DMKVMA	DMKVSP								
DMKCVTDB	000089	DMKCDB	DMKCDM	DMKCDS	DMKCFC	DMKCFG	DMKCFO	DMKCFS	DMKCFT	DMKCQH	DMKCQY	DMKCSB	DMKCSO
		DMKCSP	DMKCST	DMKCSU	DMKCSV	DMKDEF	DMKHVD	DMKMCC	DMKMCD	DMKMCH	DMKMSG	DMKNES	DMKRGA
DMKCVTDT	000040	DMKCFO	DMKCPI	DMKCQY	DMKHVC	DMKJRL	DMKLOH	DMKMIA	DMKMID	DMKMNI	DMKMSG	DMKNLE	DMKQCN
		DMKRNH	DMKRSP	DMKSEP	DMKSPL	DMKUSO	DMKVSP						
DMKCVTFP	000004	DMKCDB	DMKCDM										
DMKCVTHB	000144	DMKCDB	DMKCDM	DMKCDS	DMKCFC	DMKCFD	DMKCFG	DMKCFO	DMKCFS	DMKCPB	DMKCPS	DMKCPV	DMKCQG
		DMKCQP	DMKCSB	DMKCSO	DMKCSP	DMKCSQ	DMKCST	DMKDEF	DMKDIA	DMKGRF	DMKLNK	DMKMCC	DMKMCD
		DMKNES	DMKNET	DMKNLD	DMKNLE	DMKRGA	DMKSSP	DMKVDC					
DMKDASER	000002	DMKIOS	DMKSYM										
DMKDEFIN	000002	DMKCFC											
DMKDGDDK	000003	DMKHVC	DMKSYM										
DMKDIA	000001	DMKSYM											
DMKDIACP	000002	DMKCFC											
DMKDIADR	000006	DMKCFP	DMKDIB										
DMKDIAIR	000001	DMKIOS											
DMKDIAL	000002	DMKCFC											
DMKDIB	000001	DMKSYM											
DMKDIBSM	000004	DMKCCW	DMKVCA										
DMKDMPAA	000002	DMKCPI	DMKCPU										
DMKDMPAU	000002	DMKCFO	DMKCPI										
DMKDMPDK	000007	DMKPRG	DMKPSA	DMKSVC	DMKSYM								
DMKDMPDT	000002	DMKCPI	DMKMID										
DMKDMPDV	000003	DMKCFO	DMKCPI	DMKCQR									
DMKDMPMA	000002	DMKCPI	DMKCPU										
DMKDMPRC	000002	DMKCPI											
DMKDMPRS	000002	DMKCPS											

LABEL	COUNT	REFERENCES											
DMKDMPSA	000002	DMKCPI	DMKCPU										
DMKDMPSD	000001	DMKCPI											
DMKDMPSF	000001	DMKCPI											
DMKDMPSW	000002	DMKCFO	DMKCQR										
DMKDMPTD	000002	DMKCPI	DMKMID										
DMKDMPTR	000002	DMKCDB	DMKCDM										
DMKDRD	000001	DMKSYM											
DMKDRDDD	000002	DMKSPL											
DMKDRDER	000002	DMKHVD											
DMKDRDMP	000002	DMKHVD											
DMKDRDSY	000002	DMKHVD											
DMKDSBRD	000004	DMKCPS	DMKIOS	DMKSYM	DMKVDD								
DMKDSBSD	000005	DMKCCW	DMKCPS	DMKSYM									
DMKDSPA	000011	DMKIOS	DMKPRG	DMKPRV	DMKSYM	DMKTMR	DMKVSI						
DMKDSPAC	000002	DMKMIA	DMKSYM										
DMKDSPB	000010	DMKCFM	DMKPRG	DMKPRV	DMKSVC	DMKSYM							
DMKDSPBC	000002	DMKMIA	DMKSYM										
DMKDSPCC	000002	DMKMIA	DMKSYM										
DMKDSPCH	000148	DMKACO	DMKCCW	DMKCFM	DMKCFO	DMKCFP	DMKCNS	DMKCPB	DMKCPI	DMKCPS	DMKCPU	DMKCSB	DMKDGD
		DMKDIA	DMKDSB	DMKENT	DMKEXT	DMKGIO	DMKGRF	DMKHVC	DMKIOE	DMKIOG	DMKIOS	DMKLOC	DMKLOG
		DMKLOK	DMKMCD	DMKMCH	DMKMCT	DMKMNI	DMKMON	DMKNLD	DMKNLE	DMKPAG	DMKPGS	DMKPGT	DMKPRG
		DMKPRV	DMKPSA	DMKPTR	DMKQCN	DMKRGA	DMKRGE	DMKRNH	DMKRSE	DMKRSP	DMKSCH	DMKSEP	DMKSPL
		DMKSSS	DMKSVC	DMKSYM	DMKTAP	DMKTCS	DMKTDK	DMKTMR	DMKTRD	DMKTRK	DMKUDR	DMKUNT	DMKUSO
		DMKVAT	DMKVCA	DMKVCN	DMKVDC	DMKVDE	DMKVIO	DMKVMA	DMKVMC	DMKVSI	DMKVSP		
DMKDSPCK	000001	DMKMIA											
DMKDSPE	000003	DMKPSA	DMKSYM										
DMKDSPEC	000001	DMKSYM											
DMKDSPIT	000001	DMKMIA											
DMKDSPNP	000027	DMKATS	DMKCFO	DMKCPI	DMKCPU	DMKCPV	DMKFRE	DMKMCC	DMKMIA	DMKMNI	DMKMON	DMKPGS	DMKPTR
		DMKSCH	DMKSYM	DMKVMA									
DMKDSPPT	000001	DMKMIA											
DMKDSPQS	000001	DMKSYM											
DMKDSPRC	000002	DMKMIA	DMKSYM										
DMKDSPRQ	000004	DMKSTK	DMKSYM	DMKVSI									
DMKDSPRU	000027	DMKEXT	DMKFRE	DMKIOS	DMKMCH	DMKPRG	DMKPRV	DMKPSA	DMKSVC	DMKSYM	DMKTMR	DMKVAT	DMKVSI
DMKDSP0	000001	DMKCPI											
DMKDSP1	000001	DMKCPI											
DMKDSP2	000001	DMKCPI											
DMKEIG80	000001	DMKIOG											
DMKEMA	000001	DMKSYM											
DMKEMA00	000001	DMKERM											
DMKEMB	000001	DMKSYM											
DMKEMB00	000001	DMKERM											
DMKEMC	000001	DMKSYM											
DMKEMC00	000001	DMKERM											
DMKENTBS	000002	DMKMNI											

LABEL	COUNT	REFERENCES											
DMKENTEC	000001	DMKMNI											
DMKENTES	000002	DMKMNI											
DMKENTET	000001	DMKMNI											
DMKENTFI	000001	DMKMIA											
DMKENTKC	000001	DMKMID											
DMKENTSC	000001	DMKMNI											
DMKENTSK	000005	DMKIOS	DMKMCD										
DMKENTST	000001	DMKMNI											
DMKENTTB	000002	DMKMNI											
DMKENTTE	000002	DMKMNI											
DMKENTTI	000001	DMKMNI											
DMKENTUT	000003	DMKMCD	DMKMNI										
DMKENT62	000002	DMKMON											
DMKEPSWD	000005	DMKLNK	DMKLOG										
DMKERM	000001	DMKSYM											
DMKERMSG	000200	DMKACO	DMKALG	DMKATS	DMKBLD	DMKCDB	DMKCDM	DMKCDS	DMKCFC	DMKCFD	DMKCFG	DMKCFH	DMKCFO
		DMKCFS	DMKCFT	DMKCKS	DMKCNS	DMKCPB	DMKCPS	DMKCPV	DMKCQG	DMKCQH	DMKCQP	DMKCQR	DMKCQY
		DMKCSB	DMKCSO	DMKCSP	DMKCSQ	DMKCST	DMKCSU	DMKCSV	DMKDEF	DMKDIA	DMKDSP	DMKIOF	DMKIOG
		DMKJRL	DMKLNK	DMKLOG	DMKMCC	DMKMCD	DMKMCH	DMKMIA	DMKMID	DMKMNI	DMKMON	DMKMSG	DMKNES
		DMKNET	DMKNLD	DMKNLE	DMKRGA	DMKRNH	DMKRSE	DMKRSP	DMKSNC	DMKSSS	DMKTCS	DMKTHI	DMKTRA
		DMKUSO	DMKVCH	DMKVDA	DMKVDD	DMKVDS	DMKVMA	DMKVSP	DMKWRM				
DMKEXTSL	000003	DMKPSA	DMKSYM										
DMKEXTSP	000002	DMKPSA	DMKSYM										
DMKFCB	000001	DMKSYM											
DMKFCBLD	000002	DMKCSB											
DMKFREAP	000003	DMKCPI	DMKCPU	DMKPTR									
DMKFREE	000444	DMKACO	DMKALG	DMKAPI	DMKATS	DMKBLD	DMKBSC	DMKCCH	DMKCCW	DMKCDE	DMKCDM	DMKCDS	DMKCFC
		DMKCFD	DMKCFG	DMKCFH	DMKCFM	DMKCFO	DMKCFP	DMKCFS	DMKCKS	DMKCNS	DMKCPB	DMKCPI	DMKCPS
		DMKCPU	DMKCPV	DMKCQG	DMKCQH	DMKCQP	DMKCQR	DMKCQY	DMKCSB	DMKCSO	DMKCSP	DMKCSQ	DMKCST
		DMKCSU	DMKCSV	DMKDAS	DMKDEF	DMKDGD	DMKDIA	DMKDIB	DMKDRD	DMKDSB	DMKDSP	DMKERM	DMKEXT
		DMKGIO	DMKGRF	DMKGRT	DMKHVC	DMKHVD	DMKIOE	DMKIOF	DMKIOG	DMKIOS	DMKISM	DMKJRL	DMKLNK
		DMKLOC	DMKLOG	DMKLOK	DMKMCC	DMKMCD	DMKMCH	DMKMCT	DMKMIA	DMKMID	DMKMNI	DMKMON	DMKMSG
		DMKMSW	DMKNES	DMKNET	DMKNLD	DMKNLE	DMKPAG	DMKPGS	DMKPGT	DMKPSA	DMKPTR	DMKQCN	DMKRGA
		DMKRGB	DMKRNH	DMKRPA	DMKRSE	DMKRSP	DMKSCH	DMKSPL	DMKSSS	DMKSVC	DMKSYM	DMKTAP	DMKTCS
		DMKTDK	DMKTHI	DMKTHR	DMKTRA	DMKTRD	DMKTRK	DMKUDR	DMKUDU	DMKUNT	DMKUSO	DMKVAT	DMKVCA
		DMKVCH	DMKVCN	DMKVDA	DMKVDC	DMKVDD	DMKVDE	DMKVDR	DMKVDS	DMKVER	DMKVMA	DMKVMC	DMKVSI
		DMKVSP	DMKVSQ	DMKWRM									
DMKFREHI	000007	DMKCPI	DMKMNI	DMKSYM	DMKUSO								
DMKFRELG	000002	DMKCPI	DMKSYM										
DMKFRELO	000005	DMKCPI	DMKMNI	DMKSYM	DMKUSO								
DMKFRELS	000001	DMKSYM											
DMKFRENP	000005	DMKMIA	DMKMON	DMKSYM	DMKUSO								
DMKFRERC	000015	DMKLOG	DMKSYM	DMKVCH	DMKVDC	DMKVDS							
DMKFRERS	000003	DMKSYM	DMKUSO										
DMKFREST	000003	DMKDSP	DMKMON	DMKUNT									
DMKFRESV	000002	DMKCPI	DMKSYM										

LABEL	COUNT	REFERENCES											
DMKFRET	000432	DMKACO	DMKATS	DMKBLD	DMKBSC	DMKCCH	DMKCCW	DMKCDB	DMKCDM	DMKCDS	DMKCFD	DMKCFG	DMKCFH
		DMKCFM	DMKCFO	DMKCFP	DMKCFS	DMKCKS	DMKCNS	DMKCPB	DMKCPI	DMKCPS	DMKCPU	DMKCQG	DMKCQH
		DMKCQP	DMKCQR	DMKCQY	DMKCSB	DMKCSO	DMKCSP	DMKCSQ	DMKCST	DMKCSU	DMKCSV	DMKDAS	DMKDEF
		DMKDGD	DMKDIA	DMKDRD	DMKDSB	DMKDSP	DMKERM	DMKGIO	DMKGRF	DMKHVC	DMKHVD	DMKIOE	DMKIOF
		DMKIOG	DMKIOS	DMKJRL	DMKLNK	DMKLOC	DMKLOG	DMKLOH	DMKMCC	DMKMCD	DMKMIA	DMKMNI	DMKMON
		DMKMSG	DMKMSW	DMKNES	DMKNET	DMKNLD	DMKNLE	DMKPAG	DMKPGS	DMKPGT	DMKPSA	DMKPTR	DMKQCN
		DMKRGA	DMKRGB	DMKRNH	DMKRSE	DMKRSP	DMKSCH	DMKSEP	DMKSPL	DMKSSS	DMKSVC	DMKSYM	DMKTAP
		DMKTCS	DMKTDK	DMKTHI	DMKTMR	DMKTRA	DMKTRC	DMKTRD	DMKTRK	DMKUDR	DMKUDU	DMKUNT	DMKUSO
		DMKVAT	DMKVCA	DMKVCH	DMKVCN	DMKVDA	DMKVDC	DMKVDD	DMKVDE	DMKVDR	DMKVDS	DMKVER	DMKVIO
		DMKVMA	DMKVMC	DMKVSI	DMKVSP	DMKVSQ							
DMKFRETL	000002	DMKDSP	DMKUNT										
DMKFRETR	000007	DMKCPI	DMKPTR	DMKSYM									
DMKGIO	000001	DMKSYM											
DMKGIOEX	000002	DMKHVC											
DMKGRFEN	000002	DMKCPV	DMKSYM										
DMKGRFIC	000002	DMKQCN	DMKSYM										
DMKGRFIN	000002	DMKIOS	DMKSYM										
DMKGRFMT	000001	DMKGRT											
DMKGRFTI	000001	DMKSYM											
DMKGRTAB	000002	DMKGRF											
DMKGRTAI	000001	DMKGRF											
DMKGRTBL	000001	DMKGRF											
DMKGRTDS	000003	DMKGRF	DMKQCN	DMKVCN									
DMKGRTFD	000001	DMKGRF											
DMKGRTFM	000002	DMKGRF											
DMKGRTFO	000001	DMKGRF											
DMKGRTIN	000002	DMKGRF	DMKVCN										
DMKGRTPF	000001	DMKGRF											
DMKGRTP6	000001	DMKGRF											
DMKHVCAL	000003	DMKPRV	DMKSYM										
DMKHVCDI	000002	DMKMIA	DMKSYM										
DMKHVCPC	000001	DMKDRD											
DMKHVD	000001	DMKSYM											
DMKHVDAL	000002	DMKHVC											
DMKHVDPP	000002	DMKCPI											
DMKIOC	000001	DMKSYM											
DMKIOCVT	000004	DMKIOF											
DMKIOECC	000002	DMKCCH											
DMKIOECQ	000002	DMKIOF											
DMKIOECT	000002	DMKIOF	DMKIOG										
DMKIOEEP	000002	DMKIOF	DMKIOG										
DMKIOEES	000007	DMKIOF	DMKIOG										
DMKIOEFL	000002	DMKCPI											
DMKIOEFM	000003	DMKHVD	DMKSYM										
DMKIOEFR	000006	DMKHVD	DMKIOG										
DMKIOEHS	000004	DMKHVD	DMKIOG										

LABEL	COUNT	REFERENCES											
DMKIOEIQ	000003	DMKIOF											
DMKIOEIR	000001	DMKCFO											
DMKIOEMC	000002	DMKMCH											
DMKIOEMQ	000002	DMKIOF											
DMKIOEMX	000002	DMKIOF	DMKIOG										
DMKIOENI	000002	DMKIOF	DMKIOG										
DMKIOENQ	000001	DMKIOF											
DMKIOERN	000004	DMKRGA	DMKRNH										
DMKIOERP	000001	DMKIOF											
DMKIOERQ	000002	DMKIOF											
DMKIOERR	000007	DMKCNS	DMKGRF	DMKIOS	DMKSYM								
DMKIOESD	000004	DMKDSB	DMKRSE										
DMKIOESQ	000001	DMKIOF											
DMKIOESR	000008	DMKCPS	DMKNES	DMKNET									
DMKIOEST	000012	DMKBSC	DMKCNS	DMKDAS	DMKGRF	DMKRSE	DMKTAP						
DMKIOEVQ	000001	DMKIOF											
DMKIOEVR	000002	DMKVER											
DMKIOF	000001	DMKSYM											
DMKIOFC1	000002	DMKIOE											
DMKIOFIN	000004	DMKIOE											
DMKIOFM1	000002	DMKIOE											
DMKIOFOB	000006	DMKIOE											
DMKIOFST	000002	DMKIOE											
DMKIOFVR	000002	DMKIOE											
DMKIOG	000001	DMKSYM											
DMKIOGAP	000003	DMKAPI											
DMKIOGF1	000002	DMKIOE											
DMKIOGF2	000002	DMKIOE											
DMKIOSCT	000001	DMKMIA											
DMKIOSER	000001	DMKDSP											
DMKIOSHA	000009	DMKCFP	DMKCPS	DMKDIA	DMKRGB	DMKSYM							
DMKIOSIN	000003	DMKCPI	DMKSYM										
DMKIOSMQ	000002	DMKCPI	DMKSYM										
DMKIOSNM	000001	DMKMON											
DMKIOSQR	000091	DMKACO	DMKCFP	DMKCNS	DMKCPB	DMKCPI	DMKCPS	DMKCSB	DMKDIA	DMKDSB	DMKGRF	DMKIOG	DMKMNI
		DMKMON	DMKNLD	DMKNLE	DMKPAG	DMKRGA	DMKRGB	DMKRNH	DMKRSE	DMKRSP	DMKSEP	DMKSPL	DMKSYM
		DMKTAP	DMKTCS	DMKTDK	DMKTRK	DMKUDR	DMKVDC	DMKVDE	DMKVDR				
DMKIOSQV	000013	DMKDGD	DMKGIO	DMKSYM	DMKTRK	DMKVSI							
DMKIOSRC	000001	DMKDSP											
DMKIOSRW	000003	DMKCFP	DMKCPB	DMKVDR									
DMKISM	000001	DMKSYM											
DMKISMTR	000002	DMKCCW											
DMKJRLIL	000002	DMKLNK											
DMKJRLLO	000002	DMKLOG											
DMKJRLQU	000001	DMKCFC											
DMKJRLSE	000001	DMKCFC											

LABEL	COUNT	REFERENCES											
DMKJRLSL	000002	DMKLNK											
DMKLNK	000001	DMKSYM											
DMKLNKIN	000002	DMKCFC											
DMKLNKSB	000003	DMKLOG	DMKSYM										
DMKLNKSS	000001	DMKSSS											
DMKLOC	000001	DMKSYM											
DMKLOCK	000004	DMKLNK											
DMKLOCKD	000038	DMKCFP	DMKCKS	DMKDEF	DMKLNK	DMKTRA	DMKTRC	DMKTRD	DMKUDR	DMKUDU	DMKUSO	DMKVCH	DMKVDA
		DMKVDD											
DMKLOCKQ	000028	DMKCFP	DMKCKS	DMKDEF	DMKTRA	DMKTRC	DMKTRD	DMKUDR	DMKUDU	DMKUSO	DMKVCH	DMKVDA	DMKVDD
DMKLOCKT	000002	DMKCKS											
DMKLOG	000001	DMKSYM											
DMKLOGB	000002	DMKALG											
DMKLOGON	000004	DMKCFC											
DMKLOGOP	000003	DMKCPI	DMKSYM										
DMKLOGSS	000001	DMKSSS											
DMKLOHON	000003	DMKLOG	DMKSYM										
DMKLOKCT	000003	DMKMIA	DMKSYM										
DMKLOKDF	000025	DMKAPI	DMKBLD	DMKDSP	DMKEXT	DMKFRE	DMKIOS	DMKMCH	DMKMCT	DMKPRG	DMKPRV	DMKPSA	DMKPTR
		DMKSVC	DMKSYM	DMKTMR	DMKVAT	DMKVMA	DMKVSI						
DMKLOKDS	000017	DMKDSP	DMKMCT	DMKMIA	DMKSCH	DMKSTK	DMKSYM						
DMKLOKFR	000007	DMKFRE	DMKMCT	DMKMIA	DMKSYM								
DMKLOKPS	000007	DMKDSP	DMKMCT	DMKSYM									
DMKLOKRL	000014	DMKDSP	DMKMCT	DMKMIA	DMKSCH	DMKSYM							
DMKLOKSI	000001	DMKEXT											
DMKLOKSP	000002	DMKPSA	DMKSYM										
DMKLOKSW	000090	DMKACO	DMKALG	DMKCFO	DMKCFP	DMKCNS	DMKCPS	DMKCPU	DMKCPV	DMKCSV	DMKDIA	DMKGRF	DMKJRL
		DMKLOG	DMKLOH	DMKMIA	DMKMID	DMKMSG	DMKMSW	DMKNES	DMKNLD	DMKQCN	DMKRGA	DMKRGB	DMKRNH
		DMKSPL	DMKSYM	DMKTCS	DMKVCA	DMKVCH	DMKVDA	DMKVDD	DMKVMC				
DMKLOKSY	000057	DMKAPI	DMKCPU	DMKDSP	DMKEXT	DMKFRE	DMKIOS	DMKMCH	DMKMCT	DMKMIA	DMKPRG	DMKPRV	DMKPSA
		DMKSVC	DMKSYM	DMKTMR	DMKVAT	DMKVMA	DMKVSI						
DMKLOKTR	000011	DMKDSP	DMKMCT	DMKMIA	DMKPSA	DMKSCH	DMKSYM						
DMKLOKVM	000002	DMKPSA	DMKSYM										
DMKMCC	000001	DMKSYM											
DMKMCCCL	000008	DMKCFC	DMKMIA										
DMKMCDIN	000002	DMKMCC											
DMKMCDLI	000002	DMKMCC											
DMKMCDSE	000002	DMKMCC											
DMKMCDST	000002	DMKMCC											
DMKMCDTI	000002	DMKMCC											
DMKMCHIN	000002	DMKCPI	DMKSYM										
DMKMCHMS	000003	DMKCFO	DMKSYM										
DMKMCHSE	000003	DMKDSP	DMKSYM										
DMKMCHST	000005	DMKCCH	DMKPAG	DMKSYM									
DMKMCTAF	000001	DMKCPU											
DMKMCTMA	000005	DMKEXT	DMKSYM										

LABEL	COUNT	REFERENCES						
DMKMCTPR	000005	DMKDSP	DMKEXT	DMKSYM				
DMKMCTPT	000003	DMKMCH	DMKSYM					
DMKMCTST	000003	DMKMCH	DMKSYM					
DMKMIA	000001	DMKMON						
DMKMIACC	000006	DMKMCC	DMKMNI	DMKMON				
DMKMIADL	000002	DMKMCC						
DMKMIAEN	000004	DMKENT						
DMKMIAIN	000004	DMKENT						
DMKMIAKC	000002	DMKENT						
DMKMIAMU	000004	DMKMCC	DMKMCD					
DMKMIARO	000002	DMKMCC						
DMKMIAWO	000004	DMKMON						
DMKMIAX1	000002	DMKMON						
DMKMIAX2	000001	DMKMON						
DMKMIAY1	000001	DMKMON						
DMKMIAY2	000001	DMKMON						
DMKMIDNT	000003	DMKSCH	DMKSYM					
DMKMNI	000001	DMKSYM						
DMKMNIDK	000002	DMKMCC	DMKMIA					
DMKMNIDS	000002	DMKMIA						
DMKMNIFI	000004	DMKENT	DMKMON					
DMKMNISH	000008	DMKCPS	DMKMCD	DMKMIA				
DMKMNISP	000002	DMKMCC						
DMKMNIST	000014	DMKCPI	DMKMCC	DMKMCD	DMKMID			
DMKMNITH	000002	DMKMCC						
DMKMNITR	000002	DMKMON						
DMKMON	000001	DMKSYM						
DMKMONMI	000001	DMKMCC						
DMKMONPR	000006	DMKMNI						
DMKMONTI	000001	DMKMCC						
DMKMON00	000002	DMKMNI						
DMKMON40	000002	DMKMNI						
DMKMSG	000001	DMKSYM						
DMKMSGEC	000002	DMKCFC						
DMKMSGMS	000004	DMKCFC						
DMKMSGNH	000002	DMKCFC						
DMKMSGSM	000002	DMKCFC						
DMKMSGWN	000004	DMKCFC						
DMKMSWR	000013	DMKBSC	DMKCNS	DMKDAS	DMKGRF	DMKRSE	DMKSYM	DMKTAP
DMKNEM	000001	DMKSYM						
DMKNEMOP	000006	DMKTRC	DMKTRD					
DMKNES	000001	DMKSYM						
DMKNESDS	000002	DMKNET						
DMKNESEP	000002	DMKNET						
DMKNESHD	000002	DMKNET						
DMKNESPL	000002	DMKNET						

LABEL	COUNT	REFERENCES											
DMKNESTR	000002	DMKNET											
DMKNESWN	000002	DMKNET											
DMKNET	000001	DMKSYM											
DMKNETAE	000002	DMKCPI											
DMKNETWK	000002	DMKCFC											
DMKNLDR	000006	DMKCPI	DMKNET	DMKRNH									
DMKNLE	000001	DMKSYM											
DMKNLEMP	000004	DMKNET	DMKRNH										
DMKOPRWT	000011	DMKCKP	DMKDMP	DMKMCH	DMKMCT	DMKRSP	DMKSAV	DMKSYM					
DMKPAGCC	000001	DMKMIA											
DMKPAGHI	000001	DMKCPI											
DMKPAGIO	000009	DMKCDS	DMKPTR	DMKRPA	DMKSYM								
DMKPAGLO	000001	DMKCPI											
DMKPAGPS	000002	DMKMIA	DMKSYM										
DMKPAGQ	000002	DMKPTR											
DMKPAGQR	000001	DMKCQR											
DMKPAGSK	000001	DMKUSO											
DMKPAGST	000002	DMKCPI	DMKSYM										
DMKPAGWS	000001	DMKCPU											
DMKPER	000001	DMKSYM											
DMKPERIL	000008	DMKPRG	DMKPRV	DMKVAT									
DMKPERT	000006	DMKCFP	DMKDSP	DMKUSO									
DMKPGS	000001	DMKSYM											
DMKPGSPO	000015	DMKCFG	DMKCFP	DMKCPB	DMKDEF	DMKMCH	DMKMCT	DMKSYM	DMKUSO				
DMKPGSPP	000007	DMKCFG	DMKCFP	DMKSYM	DMKUSO								
DMKPGSPR	000006	DMKCFG	DMKCPV	DMKRPA									
DMKPGSPS	000006	DMKCFG											
DMKPGSSS	000002	DMKHVC											
DMKPGTBN	000002	DMKCPI	DMKSYM										
DMKPGTCG	000002	DMKNLE											
DMKPGTPG	000007	DMKCDS	DMKCPI	DMKPTR	DMKSYM								
DMKPGTPR	000006	DMKATS	DMKPGS	DMKPTR	DMKRPA								
DMKPGTP0	000003	DMKCPI	DMKTDK	DMKVDC									
DMKPGTP4	000003	DMKCPI	DMKTDK	DMKVDC									
DMKPGTP5	000004	DMKCPI	DMKTDK	DMKUSO	DMKVDC								
DMKPGTSD	000011	DMKDRD	DMKNLE	DMKSPL									
DMKPGTSG	000018	DMKMIA	DMKRSP	DMKSPL	DMKVSP	DMKVSQ							
DMKPGTSP	000004	DMKCPU	DMKPGS	DMKRPA									
DMKPGTSR	000002	DMKSPL											
DMKPGTTM	000004	DMKCPI	DMKSYM	DMKVDC									
DMKPGTTU	000003	DMKCKS	DMKCPI	DMKSYM									
DMKPGTT0	000004	DMKCPI	DMKTDK	DMKVDC									
DMKPGTT4	000004	DMKCPI	DMKTDK	DMKVDC									
DMKPGTT5	000004	DMKCPI	DMKTDK	DMKVDC									
DMKPGTVG	000060	DMKCKS	DMKCQH	DMKCST	DMKCSU	DMKDRD	DMKIOF	DMKIOG	DMKMCC	DMKMIA	DMKNLD	DMKNLE	DMKRSP
		DMKSEP	DMKSPL	DMKTCS	DMKUDR	DMKUDU	DMKVSP	DMKVSQ	DMKWRM				

LABEL	COUNT	REFERENCES											
DMKPGTVR	000062	DMKCKS	DMKCQH	DMKCST	DMKCSU	DMKDRD	DMKIOF	DMKIOG	DMKMIA	DMKMNI	DMKNLD	DMKNLE	DMKRSP
		DMKSEP	DMKTCS	DMKUDR	DMKUDU	DMKVSP	DMKVSQ	DMKWRM					
DMKPGT4P	000003	DMKCPI	DMKTDK	DMKVDC									
DMKPGT4T	000004	DMKCPI	DMKTDK	DMKVDC									
DMKPGT5P	000003	DMKCPI	DMKTDK	DMKVDC									
DMKPGT5T	000004	DMKCPI	DMKTDK	DMKVDC									
DMKPGT90	000002	DMKCPI	DMKVDC										
DMKPRGCT	000002	DMKMIA	DMKSYM										
DMKPRGC8	000019	DMKMCC	DMKMCD	DMKMIA	DMKMNI	DMKMON	DMKSYM						
DMKPRGIN	000003	DMKAPI	DMKCPI	DMKSYM									
DMKPRGMC	000030	DMKCPS	DMKDMP	DMKENT	DMKMCC	DMKMCD	DMKMIA	DMKMID	DMKMNI	DMKMON	DMKSYM		
DMKPRGMI	000003	DMKMCC	DMKMNI										
DMKPRGRF	000003	DMKSVC	DMKSYM										
DMKPRGSM	000017	DMKDSP	DMKHVC	DMKPRV	DMKSYM	DMKTMR	DMKVAT						
DMKPRGTI	000006	DMKMCC	DMKMCD	DMKMNI	DMKMON								
DMKPRVCD	000001	DMKMIA											
DMKPRVCE	000001	DMKMIA											
DMKPRVCH	000001	DMKMIA											
DMKPRVCP	000001	DMKMIA											
DMKPRVCS	000001	DMKMIA											
DMKPRVCT	000001	DMKMIA											
DMKPRVDI	000001	DMKMIA											
DMKPRVEK	000001	DMKMIA											
DMKPRVEP	000001	DMKMIA											
DMKPRVIK	000001	DMKMIA											
DMKPRVIP	000001	DMKMIA											
DMKPRVLC	000001	DMKMIA											
DMKPRVLG	000003	DMKPRG	DMKSYM										
DMKPRVLP	000001	DMKMIA											
DMKPRVLR	000001	DMKMIA											
DMKPRVMA	000001	DMKPSA											
DMKPRVMN	000001	DMKMIA											
DMKPRVMO	000001	DMKMIA											
DMKPRVMS	000001	DMKMIA											
DMKPRVNC	000002	DMKMIA	DMKSYM										
DMKPRVPB	000001	DMKMIA											
DMKPRVPE	000001	DMKMIA											
DMKPRVPT	000001	DMKMIA											
DMKPRVRR	000001	DMKMIA											
DMKPRVTC	000001	DMKMIA											
DMKPRVTE	000001	DMKMIA											
DMKPSA	000001	DMKLD00E											
DMKPSACC	000032	DMKCDS	DMKDGD	DMKVCN	DMKVSP								
DMKPSADU	000004	DMKAPI	DMKCLK	DMKCPI	DMKSYM								
DMKPSAER	000001	DMKEXT											
DMKPSAEX	000002	DMKCPI	DMKSYM										

LABEL	COUNT	REFERENCES											
DMKPSAFC	000002	DMKVMC											
DMKPSAFP	000012	DMKPRV	DMKTMR										
DMKPSANX	000002	DMKMIA											
DMKPSARR	000002	DMKTRC											
DMKPSARS	000006	DMKTRC											
DMKPSARX	000004	DMKTRC											
DMKPSASC	000030	DMKCDS	DMKCFD	DMKDGD	DMKTRC	DMKVCN	DMKVMC	DMKVSP					
DMKPSASP	000026	DMKDRD	DMKHVC	DMKHVD	DMKPRV	DMKTMR	DMKTRC	DMKVMC					
DMKPTRAN	000201	DMKAPI	DMKATS	DMKBLD	DMKCCH	DMKCCW	DMKCDE	DMKCDM	DMKCDS	DMKCFD	DMKCFG	DMKCFH	DMKCKS
		DMKCNS	DMKCPB	DMKCPI	DMKCPS	DMKCPU	DMKCPV	DMKCSO	DMKDGD	DMKDRD	DMKDSP	DMKERM	DMKGIO
		DMKGRF	DMKGRT	DMKHVC	DMKHVD	DMKIOF	DMKIOG	DMKISM	DMKMCC	DMKMIA	DMKMON	DMKNLD	DMKNLE
		DMKPGS	DMKPRG	DMKPRV	DMKPSA	DMKQCN	DMKRGA	DMKRGB	DMKRPA	DMKRSP	DMKSEP	DMKSNC	DMKSPL
		DMKSSS	DMKSVC	DMKSYM	DMKTCS	DMKTMR	DMKTRC	DMKTRK	DMKUDR	DMKUDU	DMKVAT	DMKVCH	DMKVCN
		DMKVDR	DMKVER	DMKVIO	DMKVMC	DMKVSI	DMKVSP	DMKVSQ	DMKWRM				
DMKPTRCP	000002	DMKCPI											
DMKPTRCS	000001	DMKMIA											
DMKPTRCT	000001	DMKSYM											
DMKPTRFA	000001	DMKCPI											
DMKPTRFC	000002	DMKMIA	DMKSYM										
DMKPTRFD	000001	DMKDSP											
DMKPTRFE	000001	DMKDSP											
DMKPTRFF	000003	DMKCQR	DMKMIA	DMKSYM									
DMKPTRFN	000002	DMKCPI	DMKMIA										
DMKPTRFP	000001	DMKDSP											
DMKPTRFR	000014	DMKCCW	DMKCPU	DMKDGD	DMKFRE								
DMKPTRFT	000028	DMKATS	DMKCPU	DMKDGD	DMKFRE	DMKMCH	DMKPGS	DMKRPA	DMKUDR	DMKUNT	DMKVMA		
DMKPTRF0	000001	DMKMIA											
DMKPTRF1	000001	DMKCPI											
DMKPTRLK	000016	DMKACO	DMKCCW	DMKCPI	DMKCPS	DMKIOG	DMKMNI	DMKPSA	DMKSPL	DMKTRC	DMKUSO		
DMKPTRPL	000001	DMKUNT											
DMKPTRPR	000002	DMKMIA	DMKSYM										
DMKPTRPW	000016	DMKATS	DMKCFP	DMKCPU	DMKPGS	DMKUSO	DMKVAT	DMKVDR					
DMKPTRRC	000009	DMKATS	DMKCFO	DMKCPU	DMKDSP	DMKMIA	DMKPGS	DMKSYM	DMKVMA				
DMKPTRRF	000001	DMKMIA											
DMKPTRRL	000003	DMKCFO	DMKSCH	DMKUSO									
DMKPTRRM	000001	DMKCPI											
DMKPTRRQ	000002	DMKPAG	DMKSYM										
DMKPTRRS	000005	DMKPGS	DMKSCH										
DMKPTRRU	000002	DMKCFO	DMKUSO										
DMKPTRSC	000008	DMKATS	DMKCPU	DMKMIA	DMKPGS	DMKSYM	DMKVMA						
DMKPTRSS	000003	DMKCQR	DMKMIA	DMKSYM									
DMKPTRSW	000001	DMKMIA											
DMKPTRUC	000003	DMKATS	DMKSCH										
DMKPTRUL	000123	DMKACO	DMKAPI	DMKATS	DMKCCW	DMKCFG	DMKCFH	DMKCKS	DMKCPI	DMKCPS	DMKCPU	DMKCPV	DMKCSB
		DMKDGD	DMKIOF	DMKIOG	DMKISM	DMKMNI	DMKMON	DMKNLD	DMKNLE	DMKPGS	DMKRPA	DMKSEP	DMKSNC
		DMKSPL	DMKSVC	DMKTAP	DMKTRC	DMKUNT	DMKUSC	DMKVIO	DMKVMA	DMKVMC	DMKVSQ		

LABEL	COUNT	REFERENCES											
DMKPTRWQ	000004	DMKCDS	DMKPAG	DMKRPA	DMKSYM								
DMKPTRXX	000002	DMKPGT											
DMKQCNCL	000035	DMKCNS	DMKDIA	DMKGRF	DMKNES	DMKNLD	DMKPSA	DMKRGA	DMKRNH	DMKSYM			
DMKQCNET	000023	DMKCNS	DMKGRF	DMKRGA	DMKRGB	DMKRNH	DMKSYM						
DMKQCNFT	000001	DMKFRE											
DMKQCNRD	000031	DMKACO	DMKCFH	DMKCFM	DMKCFO	DMKCPI	DMKLNK	DMKMSG	DMKMSW	DMKNLD	DMKNLE	DMKSYM	DMKVCN
DMKQCNSY	000007	DMKCFP	DMKCKS	DMKLOG	DMKSYM								
DMKQCNTO	000017	DMKCNS	DMKGRF	DMKNES	DMKNLD	DMKRGA	DMKRNH	DMKSYM					
DMKQCNWT	000266	DMKACO	DMKAPI	DMKBLD	DMKCCH	DMKCDB	DMKCDM	DMKCDS	DMKCFC	DMKCFD	DMKCFH	DMKCFM	DMKCFO
		DMKCFS	DMKCLK	DMKCPB	DMKCPI	DMKCPS	DMKCPU	DMKCPV	DMKCQG	DMKCQH	DMKCQP	DMKCQR	DMKCQY
		DMKCSB	DMKCSO	DMKCSU	DMKCSV	DMKDAS	DMKDEF	DMKDIA	DMKDSE	DMKERM	DMKGRF	DMKJRL	DMKLNK
		DMKLOH	DMKMCC	DMKMCD	DMKMCH	DMKMCT	DMKMIA	DMKMID	DMKMNI	DMKMSG	DMKMSW	DMKNES	DMKNET
		DMKNLD	DMKNLE	DMKPGT	DMKPRG	DMKPSA	DMKPTR	DMKRGA	DMKRNH	DMKRSP	DMKSPL	DMKSVC	DMKSYM
		DMKTHI	DMKTRA	DMKTRC	DMKTRD	DMKUDR	DMKUSC	DMKVCA	DMKVCH	DMKVCN	DMKVDA	DMKVDD	DMKVDR
		DMKVER	DMKWRM										
DMKQNTBL	000005	DMKCKS	DMKCSO	DMKHVD	DMKTCS	DMKWRM							
DMKRGAIN	000003	DMKIOS	DMKRGB	DMKSYM									
DMKRGBEN	000011	DMKNES	DMKNET	DMKRGA	DMKSYM								
DMKRGBIC	000004	DMKQCN	DMKRGA	DMKSYM									
DMKRGBMT	000001	DMKRGA											
DMKRGBSN	000001	DMKRGA											
DMKRIOCC	000001	DMKPSA											
DMKRIOCH	000003	DMKPSA	DMKSSP	DMKSYM									
DMKRIOCN	000007	DMKCPI	DMKGRF	DMKIOS	DMKOPR	DMKSSP	DMKSYM						
DMKRIOCT	000003	DMKIOS	DMKPSA										
DMKRIOCU	000007	DMKIOS	DMKPSA	DMKSSP	DMKSYM								
DMKRIODC	000001	DMKPSA											
DMKRIODV	000010	DMKDMP	DMKIOG	DMKOPR	DMKPSA	DMKSSP	DMKSYM						
DMKRIOPR	000006	DMKCQR	DMKDMP	DMKPSA	DMKSSP	DMKSYM							
DMKRIOPU	000003	DMKPSA	DMKSSP	DMKSYM									
DMKRIORD	000003	DMKPSA	DMKSSP	DMKSYM									
DMKRIORN	000015	DMKBLD	DMKCPI	DMKCQP	DMKDIA	DMKNES	DMKNET	DMKRNH	DMKSYM				
DMKRIOSP	000001	DMKIOG											
DMKRIOUC	000001	DMKPSA											
DMKRNH	000001	DMKCPV											
DMKRNHCT	000001	DMKSYM											
DMKRNHIC	000002	DMKQCN	DMKSYM										
DMKRNHIN	000006	DMKIOS	DMKNLD	DMKNLE	DMKSYM								
DMKRNHND	000029	DMKDIA	DMKNES	DMKNET	DMKPSA	DMKSYM							
DMKRNHTG	000001	DMKSYM											
DMKRNHTR	000004	DMKNES	DMKSYM										
DMKRNTBL	000002	DMKNLD	DMKSNC										
DMKRPAGT	000136	DMKCFG	DMKCFH	DMKCKS	DMKCQH	DMKCST	DMKCSU	DMKDRD	DMKHVD	DMKIOF	DMKIOG	DMKMIA	DMKNLD
		DMKRSP	DMKSPL	DMKSYM	DMKTCS	DMKUDR	DMKUDU	DMKVSP	DMKVSQ	DMKWRM			
DMKRPAPT	000064	DMKCFH	DMKCKS	DMKCPI	DMKCST	DMKCSU	DMKHVD	DMKIOF	DMKIOG	DMKMIA	DMKNLE	DMKRSP	DMKSNC
		DMKSPL	DMKSYM	DMKUDU	DMKVSP	DMKVSQ	DMKWRM						

LABEL	COUNT	REFERENCES											
DMKRSE	000001	DMKSYM											
DMKRSERR	000002	DMKRSP											
DMKRSESD	000006	DMKCCW	DMKCPS										
DMKRSPAC	000004	DMKCKP	DMKPSA	DMKSYM	DMKWRM								
DMKRSPCV	000005	DMKCKP	DMKSYM	DMKWRM									
DMKRSPDL	000011	DMKCKP	DMKSPL	DMKSYM	DMKWRM								
DMKRSPER	000002	DMKIOS	DMKSYM										
DMKRSPEX	000005	DMKACO	DMKCSO	DMKIOS	DMKSPL	DMKSYM							
DMKRSPHQ	000009	DMKCKP	DMKCKS	DMKCQR	DMKCSQ	DMKSPL	DMKSYM	DMKWRM					
DMKRSPID	000009	DMKCKP	DMKCKS	DMKDMP	DMKMIA	DMKNLE	DMKSPL	DMKSYM	DMKWRM				
DMKRSPMN	000005	DMKCKP	DMKMCC	DMKMIA									
DMKRSPPR	000008	DMKCKP	DMKCQR	DMKPSA	DMKRSE	DMKSYM	DMKWRM						
DMKRSPPU	000006	DMKCKP	DMKCQR	DMKPSA	DMKSYM	DMKWRM							
DMKRSPRD	000008	DMKCKP	DMKCQR	DMKDMP	DMKPSA	DMKSYM	DMKWRM						
DMKRSPUR	000003	DMKCQP	DMKSYM										
DMKRSP83	000002	DMKRSE	DMKSYM										
DMKSAV	000004	DMKCKP	DMKCPI										
DMKSAVRS	000002	DMKCKP											
DMKSCHAE	000001	DMKSYM											
DMKSCHAL	000002	DMKMON	DMKSYM										
DMKSCHAP	000003	DMKCFO	DMKSYM										
DMKSCHAU	000004	DMKCFO	DMKSYM	DMKUSO									
DMKSCHCA	000002	DMKCPU	DMKTHI										
DMKSCHCO	000001	DMKTHI											
DMKSCHCP	000002	DMKBLD	DMKSYM										
DMKSCHCT	000002	DMKMIA	DMKSYM										
DMKSCHCU	000001	DMKTHI											
DMKSCHDL	000049	DMKACO	DMKALG	DMKCFO	DMKDSP	DMKIOS	DMKPTR	DMKQCN	DMKRPA	DMKSYM	DMKUSO	DMKVCA	DMKVMC
		DMKVSI											
DMKSCHEL	000004	DMKTHI											
DMKSCHIB	000001	DMKSYM											
DMKSCHLI	000002	DMKCPI	DMKTHI										
DMKSCHMD	000002	DMKCPI	DMKSYM										
DMKSCHN1	000007	DMKMIA	DMKMON	DMKPTR	DMKSYM	DMKTMR							
DMKSCHN2	000006	DMKMIA	DMKPTR	DMKSYM	DMKTMR								
DMKSCHPB	000001	DMKSYM											
DMKSCHPD	000001	DMKSYM											
DMKSCHPG	000002	DMKCQR	DMKSYM										
DMKSCHPU	000004	DMKCPV	DMKMIA	DMKMON	DMKSYM								
DMKSCHQ1	000004	DMKCFO	DMKCPI	DMKMON	DMKSYM								
DMKSCHQ2	000003	DMKCFO	DMKCPI	DMKSYM									
DMKSCHRL	000005	DMKDSP	DMKSYM	DMKTHI									
DMKSCHRT	000046	DMKCFC	DMKCFM	DMKCFP	DMKCFS	DMKDIA	DMKGRF	DMKLOG	DMKMCC	DMKMCD	DMKMNI	DMKQCN	DMKRGA
		DMKRGB	DMKSSS	DMKSYM	DMKTMR	DMKUSO							
DMKSCHSC	000001	DMKTHI											
DMKSCHST	000031	DMKCFC	DMKCPI	DMKENT	DMKGRF	DMKMCC	DMKMID	DMKMNI	DMKMON	DMKQCN	DMKRGA	DMKSSS	DMKSYM

LABEL	COUNT	REFERENCES											
		DMKTMR											
DMKSCHS1	000001	DMKTHI											
DMKSCHS2	000001	DMKTHI											
DMKSCHTI	000002	DMKCPI	DMKSYM										
DMKSCHTQ	000005	DMKCPU	DMKDSP	DMKPSA	DMKSYM								
DMKSCHUB	000001	DMKSYM											
DMKSCHW1	000003	DMKMON	DMKSYM										
DMKSCHW2	000003	DMKMON	DMKSYM										
DMKSCH80	000003	DMKCFS	DMKLOG	DMKSYM									
DMKSCNAU	000088	DMKCFD	DMKCFO	DMKCFS	DMKCPV	DMKCQG	DMKCQP	DMKCQR	DMKCQY	DMKCSU	DMKCSV	DMKDIA	DMKJRL
		DMKLNK	DMKLOG	DMKMIA	DMKMSG	DMKRNH	DMKSPL	DMKSSS	DMKTCS	DMKUDU	DMKUSO	DMKVCH	DMKVDA
		DMKVDC	DMKVDD	DMKVMC									
DMKSCNFD	000456	DMKCDB	DMKCDM	DMKCDS	DMKCFC	DMKCFD	DMKCFG	DMKCFH	DMKCFM	DMKCFO	DMKCFS	DMKCFT	DMKCPB
		DMKCPS	DMKCPV	DMKCQG	DMKCQH	DMKCQP	DMKCQR	DMKCQY	DMKCSB	DMKCSO	DMKCSP	DMKCSQ	DMKCST
		DMKCSU	DMKCSV	DMKDEF	DMKDIA	DMKJRL	DMKLNK	DMKLOG	DMKMCC	DMKMCD	DMKMCH	DMKMIA	DMKMNI
		DMKMSG	DMKNES	DMKNET	DMKNLD	DMKNLE	DMKRSP	DMKTHI	DMKTRA	DMKUSO	DMKVDC		
DMKSCNLI	000004	DMKLNK	DMKVDS										
DMKSCNNP	000002	DMKCPS											
DMKSCNRA	000012	DMKCFO	DMKCPI	DMKCPS	DMKCQP	DMKVCH							
DMKSCNRD	000098	DMKBLD	DMKCKS	DMKCNS	DMKCPI	DMKCPV	DMKCQG	DMKCQP	DMKCQR	DMKCQY	DMKDIA	DMKDMP	DMKGRF
		DMKJRL	DMKLOG	DMKLOH	DMKNES	DMKNET	DMKNLD	DMKNLE	DMKPAG	DMKPSA	DMKQCN	DMKRGA	DMKRSP
		DMKSEP	DMKTCS	DMKTHI	DMKTRD	DMKUSO	DMKVDA	DMKVDD	DMKVDR	DMKVDS	DMKVER		
DMKSCNRN	000034	DMKCPS	DMKCPV	DMKCQG	DMKCQP	DMKDIA	DMKMSW	DMKQCN	DMKRSP	DMKTRD	DMKUSO	DMKVDA	DMKVDD
		DMKVDR											
DMKSCNRU	000157	DMKCCH	DMKCCW	DMKCFD	DMKCFO	DMKCKS	DMKCNS	DMKCPI	DMKCPS	DMKCPV	DMKCQP	DMKCQR	DMKCSB
		DMKCSO	DMKDEF	DMKDIA	DMKDMP	DMKDSB	DMKGRF	DMKHVD	DMKIOG	DMKIOS	DMKLOG	DMKMCC	DMKMCD
		DMKNES	DMKNET	DMKNLD	DMKNLE	DMKRGA	DMKRNH	DMKRSP	DMKSSS	DMKVCH	DMKVDA	DMKVDC	DMKVDD
		DMKVDE	DMKVDS	DMKWRM									
DMKSCNVD	000036	DMKALG	DMKCCW	DMKCFP	DMKCFT	DMKCPS	DMKCQP	DMKCST	DMKDEF	DMKDIA	DMKGIO	DMKHVD	DMKLOG
		DMKUNT	DMKUSO	DMKVSP									
DMKSCNVN	000018	DMKCPS	DMKCQG	DMKCST	DMKDEF	DMKLNK	DMKLOG	DMKTRD	DMKVDD				
DMKSCNVS	000034	DMKATS	DMKCFG	DMKCFH	DMKCPI	DMKCPU	DMKHVD	DMKLNK	DMKLOG	DMKNLD	DMKSNC	DMKSSS	DMKTCS
		DMKVDA											
DMKSCNVU	000135	DMKCFD	DMKCFG	DMKCFH	DMKCPB	DMKCPI	DMKCPS	DMKCQG	DMKCQP	DMKCSB	DMKCSP	DMKCSQ	DMKCST
		DMKDEF	DMKDGD	DMKDIA	DMKDRD	DMKDSP	DMKGIC	DMKHVC	DMKHVD	DMKLNK	DMKLOG	DMKNLD	DMKPRV
		DMKSSS	DMKTHI	DMKTRD	DMKVCA	DMKVCH	DMKVCN	DMKVDA	DMKVDC	DMKVDD	DMKVDS	DMKVER	DMKVIO
		DMKVSI	DMKVSP										
DMKSEP	000001	DMKSYM											
DMKSEPSP	000002	DMKRSP											
DMKSEV70	000002	DMKIOG	DMKSYM										
DMKSIX60	000002	DMKIOG	DMKSYM										
DMKSNCP	000002	DMKHVD											
DMKSNTBL	000004	DMKATS	DMKCFG	DMKCFH	DMKCPU								
DMKSPL	000001	DMKSYM											
DMKSPLCR	000002	DMKRSP											
DMKSPLCV	000004	DMKVSP	DMKVSQ										

LABEL	COUNT	REFERENCES											
DMKSPLDL	000022	DMKCPS	DMKCSO	DMKCSV	DMKMIA	DMKRSP	DMKVDS	DMKVSP	DMKVSQ				
DMKSPLOR	000002	DMKRSP											
DMKSPLOV	000002	DMKVSP											
DMKSPLSP	000002	DMKMIA											
DMKSSSAS	000002	DMKVDA											
DMKSSSCF	000002	DMKCFP											
DMKSSSDE	000002	DMKDEF											
DMKSSSEN	000001	DMKDSB											
DMKSSSHV	000002	DMKHVC											
DMKSSSI1	000001	DMKIOS											
DMKSSSI2	000001	DMKIOS											
DMKSSSLN	000002	DMKLNK											
DMKSSSL1	000002	DMKLOG											
DMKSSSL2	000002	DMKLOG											
DMKSSSL3	000002	DMKLOG											
DMKSSSMQ	000007	DMKCPB	DMKDGD	DMKDSB	DMKLNK	DMKLOG	DMKVDA	DMKVSI					
DMKSSSUS	000002	DMKUSO											
DMKSSSVA	000002	DMKVDA											
DMKSSSVM	000001	DMKUSO											
DMKSTKCP	000137	DMKACO	DMKALG	DMKCFM	DMKCFP	DMKCPB	DMKCPS	DMKCPV	DMKDIA	DMKDSB	DMKGRF	DMKIOE	DMKIOF
		DMKLNK	DMKLOG	DMKMCC	DMKMIA	DMKMID	DMKMNI	DMKMON	DMKPAG	DMKPGS	DMKPGT	DMKPTR	DMKQCN
		DMKRGB	DMKRNH	DMKRSP	DMKSPL	DMKSSS	DMKSYM	DMKTRD	DMKUNT	DMKUSO	DMKVCA	DMKVDA	DMKVMA
		DMKVMC	DMKVSP										
DMKSTKDE	000015	DMKDSP	DMKPRG	DMKPRV	DMKSVC	DMKSYM	DMKTMR	DMKVAT	DMKVSI				
DMKSTKIO	000071	DMKACO	DMKCFP	DMKCPS	DMKCSO	DMKCSP	DMKCSU	DMKCSV	DMKDIA	DMKDIE	DMKIOS	DMKNLD	DMKNLE
		DMKPAG	DMKPSA	DMKSPL	DMKSYM	DMKTMR	DMKUNT	DMKVCA	DMKVDD	DMKVSI			
DMKSTKLF	000009	DMKFRE	DMKIOS	DMKSYM									
DMKSTKMP	000029	DMKCNS	DMKCPU	DMKEXT	DMKGRF	DMKIOS	DMKLOC	DMKMCT	DMKPAG	DMKPSA	DMKPTR	DMKRGA	DMKSYM
		DMKVMA	DMKVSI										
DMKSTKOP	000017	DMKCFO	DMKCPU	DMKLOC	DMKMCD	DMKMCH	DMKPAG	DMKPTR	DMKSYM	DMKVSI			
DMKSTKSW	000006	DMKLOK	DMKSVC										
DMKSVCHI	000001	DMKCPI											
DMKSVCIN	000003	DMKAPI	DMKCPI	DMKSYM									
DMKSVCLO	000001	DMKCPI											
DMKSVCNS	000002	DMKCPI	DMKPSA										
DMKSYM	000002	DMKCPI	DMKDRD										
DMKSYMTB	000001	DMKCPI											
DMKSYSAP	000011	DMKATS	DMKCDB	DMKCDM	DMKCDS	DMKCFG	DMKCPI	DMKPGS					
DMKSYSAT	000015	DMKCPI	DMKENT	DMKMCC	DMKMCD	DMKMIA	DMKMID	DMKMNI	DMKMON				
DMKSYSBF	000002	DMKMIA											
DMKSYSCD	000001	DMKCFT											
DMKSYSCH	000003	DMKCKS	DMKDMP										
DMKSYSCK	000022	DMKACO	DMKCKP	DMKDIA	DMKDMP	DMKLOH	DMKVDS						
DMKSYSCL	000002	DMKMIA	DMKMNI										
DMKSYSCN	000006	DMKCKS											
DMKSYSCS	000007	DMKCCW	DMKDMP	DMKFRE	DMKPSA	DMKPTR	DMKSYM						

LABEL	COUNT	REFERENCES											
DMKSYSCT	000003	DMKHVD	DMKIOG										
DMKSYSDT	000004	DMKCFO	DMKCKP	DMKLOH	DMKWRM								
DMKSYSDU	000002	DMKCPI	DMKNLE										
DMKSYSDW	000009	DMKCFO	DMKCPI	DMKCQY	DMKLOH	DMKMID	DMKUSO						
DMKSYSEN	000004	DMKMCC	DMKMIA										
DMKSYSER	000007	DMKHVD	DMKIOF	DMKIOG									
DMKSYSES	000001	DMKCFT											
DMKSYSFP	000001	DMKCPI											
DMKSYSJR	000014	DMKALG	DMKJRL	DMKLNK	DMKLOG								
DMKSYSLB	000002	DMKLOC											
DMKSYSLC	000002	DMKPSA	DMKSYM										
DMKSYSLD	000001	DMKCFT											
DMKSYSLE	000002	DMKBLD	DMKCFT										
DMKSYSLG	000007	DMKCFO	DMKCKP	DMKCQY	DMKLOH	DMKWRM							
DMKSYSLL	000003	DMKBLD											
DMKSYSLW	000003	DMKCFO	DMKLOH										
DMKSYSMA	000001	DMKLOG											
DMKSYSMU	000004	DMKLOH											
DMKSYSMX	000005	DMKMCD	DMKMNI	DMKMON									
DMKSYSND	000004	DMKCQY	DMKDIA	DMKMIA									
DMKSYSNM	000014	DMKCQY	DMKGRF	DMKLOG	DMKLOH	DMKMIA	DMKMON	DMKQCN	DMKUSO				
DMKSYSNU	000003	DMKCPI	DMKSAV	DMKSSP									
DMKSYSOC	000012	DMKCKP	DMKCKS	DMKCPI	DMKMON	DMKRSP	DMKSPL	DMKSYM	DMKUDR	DMKVDA			
DMKSYSOP	000002	DMKPSA	DMKSYM										
DMKSYSOW	000026	DMKATS	DMKCKP	DMKCKS	DMKCPI	DMKCPU	DMKDRD	DMKMON	DMKPAG	DMKPGS	DMKPGT	DMKPTR	DMKRSP
		DMKSPL	DMKSYM	DMKUDR	DMKVDA	DMKVDC	DMKWRM						
DMKSYSPL	000004	DMKUDR	DMKUDU										
DMKSYSRM	000034	DMKCCH	DMKCCW	DMKCDB	DMKCDM	DMKCDS	DMKCKP	DMKCPI	DMKCQP	DMKDIE	DMKDMP	DMKERM	DMKFRE
		DMKHVD	DMKMNI	DMKPTR	DMKRSP	DMKSYM	DMKTRD	DMKUNT	DMKVCA				
DMKSYSRS	000002	DMKSAV	DMKSYM										
DMKSYSRV	000010	DMKCFO	DMKCPI	DMKCQP	DMKDMP	DMKMNI	DMKSYM						
DMKSYSTE	000008	DMKCPI	DMKMCD	DMKMID	DMKMNI								
DMKSYSTI	000006	DMKCPI	DMKCQY	DMKLOH	DMKMID	DMKUSO							
DMKSYSTM	000002	DMKCFO	DMKLOH										
DMKSYSTP	000004	DMKCKP	DMKRSP	DMKSAV	DMKSSP								
DMKSYSTR	000001	DMKCPI											
DMKSYSTS	000008	DMKCPI	DMKMCD	DMKMID	DMKMNI								
DMKSYSTZ	000005	DMKCPI	DMKIOE	DMKIOG	DMKSAV								
DMKSYSUD	000011	DMKCPI	DMKUDR	DMKUDU									
DMKSYSUR	000009	DMKCPI	DMKMIA	DMKMNI									
DMKSYSVL	000004	DMKCPI	DMKSAV	DMKSYM									
DMKSYSVM	000004	DMKDMP	DMKPSA	DMKSYM									
DMKSYSWM	000003	DMKCKP	DMKRSP	DMKWRM									
DMKTAPER	000002	DMKIOS	DMKSYM										
DMKTAPRL	000001	DMKCPS											
DMKTBL	000001	DMKSYM											

LABEL	COUNT	REFERENCES					
DMKTBLCI	000001	DMKCNS					
DMKTBLCO	000001	DMKCNS					
DMKTBLGL	000002	DMKGRF					
DMKTBLGR	000003	DMKCFT	DMKRGA	DMKRGB			
DMKTBLPI	000001	DMKCNS					
DMKTBLPO	000001	DMKCNS					
DMKTBLRG	000001	DMKRGB					
DMKTBLTI	000001	DMKCNS					
DMKTBLTO	000001	DMKCNS					
DMKTBLUP	000004	DMKCNS	DMKGRF	DMKRGA	DMKVCN		
DMKTBM	000001	DMKSYM					
DMKTBMMI	000001	DMKCNS					
DMKTBMMO	000001	DMKCNS					
DMKTBMNI	000001	DMKCNS					
DMKTBMNO	000001	DMKCNS					
DMKTBMTI	000002	DMKGRF	DMKRGA				
DMKTBMTO	000002	DMKGRF	DMKRGB				
DMKTBMZI	000002	DMKGRF	DMKRGA				
DMKTBMZO	000002	DMKGRF	DMKRGB				
DMKTCS	000001	DMKSYM					
DMKTCSCO	000006	DMKRSP					
DMKTCSET	000002	DMKRSP					
DMKTCSSP	000002	DMKSEP					
DMKTDK	000001	DMKSYM					
DMKTDKGT	000002	DMKVDS					
DMKTDKRL	000002	DMKVDR					
DMKTHI	000001	DMKSYM					
DMKTHIEN	000002	DMKCFC					
DMKTMR	000001	DMKPRV					
DMKTMRCC	000001	DMKPRV					
DMKTMRCK	000001	DMKBLD					
DMKTMRPT	000013	DMKACO	DMKCKP	DMKHVC	DMKRSP	DMKTHI	DMKVSP
DMKTMRSN	000001	DMKPSA					
DMKTMRSP	000001	DMKPRV					
DMKTMRTN	000003	DMKPRV	DMKSYM				
DMKTMRVT	000002	DMKPSA					
DMKTRA	000001	DMKSYM					
DMKTRACE	000002	DMKCFC					
DMKTRCEX	000002	DMKDSP					
DMKTRCIO	000002	DMKDSP					
DMKTRCIT	000010	DMKCDS	DMKCFC	DMKDSP	DMKSVC	DMKTRA	
DMKTRCND	000002	DMKUSO					
DMKTRCPB	000016	DMKCDS	DMKCFC	DMKCFP	DMKPRV	DMKSVC	DMKTRA
DMKTRCPG	000004	DMKDSP	DMKPRG				
DMKTRCPV	000002	DMKPRV					
DMKTRCSV	000002	DMKSVC					

LABEL	COUNT	REFERENCES											
DMKTRCSW	000002	DMKVIO											
DMKTRDSI	000008	DMKIOS	DMKVCA	DMKVIO	DMKVSI								
DMKTRDWT	000002	DMKVIO											
DMKTRKFP	000005	DMKSYM	DMKUNT	DMKVIO									
DMKTRKIN	000003	DMKDAS	DMKSYM										
DMKTRKVA	000003	DMKCCW	DMKSYM										
DMKTRM	000001	DMKSYM											
DMKTRMID	000002	DMKCNS											
DMKUCB	000001	DMKSYM											
DMKUCBLD	000001	DMKCSB											
DMKUCC	000001	DMKSYM											
DMKUCCLD	000001	DMKCSB											
DMKUCS	000001	DMKSYM											
DMKUCSLD	000001	DMKCSB											
DMKUDR	000001	DMKSYM											
DMKUDRBV	000002	DMKCPI											
DMKUDRDS	000003	DMKHVD	DMKSYM										
DMKUDRFD	000004	DMKLNK											
DMKUDRFU	000034	DMKCFS	DMKCPI	DMKCSP	DMKCSQ	DMKCSU	DMKCSV	DMKDEF	DMKHVD	DMKLNK	DMKLOG	DMKMNI	DMKRSP
		DMKSPL											
DMKUDRMD	000010	DMKCFS	DMKDEF	DMKHVD	DMKLOG	DMKSPL							
DMKUDRRD	000002	DMKLOG											
DMKUDRRV	000018	DMKCFS	DMKDEF	DMKHVD	DMKLNK	DMKLOG	DMKSPL						
DMKUDUMN	000002	DMKHVD											
DMKUNTFR	000018	DMKCCW	DMKCFP	DMKCPI	DMKGIO	DMKHVC	DMKSYM	DMKVIO					
DMKUNTIS	000003	DMKISM	DMKSYM										
DMKUNTRN	000008	DMKCPI	DMKGIO	DMKSYM	DMKTRK	DMKVIO							
DMKUNTRS	000003	DMKCCW	DMKSYM										
DMKUSO	000001	DMKSYM											
DMKUSODS	000002	DMKCFC											
DMKUSOFF	000002	DMKDSP											
DMKUSOFL	000002	DMKCFC											
DMKUSOLG	000004	DMKCFC											
DMKVATAB	000015	DMKCDB	DMKCDM	DMKCDS	DMKDSP	DMKPRV	DMKSYM						
DMKVATAT	000002	DMKDSP	DMKPRV										
DMKVATBC	000011	DMKCDS	DMKCFP	DMKCPB	DMKDSP	DMKSYM	DMKUSO						
DMKVATEX	000007	DMKDSP	DMKPRV	DMKSYM	DMKTMR								
DMKVATLA	000003	DMKPRV	DMKSYM										
DMKVATMD	000009	DMKCDS	DMKCFG	DMKCPB	DMKDSP	DMKSYM							
DMKVATPF	000002	DMKPRG											
DMKVATPX	000003	DMKPRG	DMKSYM										
DMKVATRN	000013	DMKPRV	DMKSYM	DMKTMR	DMKTRC	DMKTRD	DMKVER						
DMKVATSX	000003	DMKPRG	DMKSYM										
DMKVATZP	000001	DMKCPI											
DMKVATZS	000001	DMKCPI											
DMKVCA	000001	DMKSYM											

LABEL	COUNT	REFERENCES											
DMKVCARD	000002	DMKCFP											
DMKVCARS	000006	DMKDEF	DMKDIA	DMKVDR									
DMKVCASH	000002	DMKVSI											
DMKVCAST	000002	DMKVSI											
DMKVCATS	000002	DMKVSI											
DMKVCH	000001	DMKSYM											
DMKVCHDC	000004	DMKVDA	DMKVDD										
DMKVCNEX	000003	DMKSYM	DMKVSI										
DMKVCNFT	000002	DMKFRE	DMKSYM										
DMKVDA	000001	DMKSYM											
DMKVDAAT	000002	DMKCFC											
DMKVDAS1	000001	DMKSSS											
DMKVDAS2	000001	DMKSSS											
DMKVDC	000001	DMKSYM											
DMKVDCAL	000002	DMKVDA											
DMKVDCPS	000002	DMKVDA											
DMKVDCSC	000004	DMKVDA	DMKVDD										
DMKVDD	000001	DMKSYM											
DMKVDDDE	000002	DMKCFC											
DMKVDE	000001	DMKSYM											
DMKVDEDC	000004	DMKVDA											
DMKVDR	000001	DMKSYM											
DMKVDREL	000010	DMKCFP	DMKNLD	DMKUSO	DMKVCH	DMKVDD							
DMKVDSAT	000009	DMKLOG	DMKVCH	DMKVDA									
DMKVDSDF	000006	DMKDEF	DMKLOG										
DMKVDSLK	000002	DMKLNK											
DMKVER	000001	DMKSYM											
DMKVERD	000001	DMKSVC											
DMKVERO	000001	DMKSVC											
DMKVIOC1	000002	DMKVSI											
DMKVIOIN	000013	DMKCFP	DMKCSP	DMKCSU	DMKCSV	DMKDIA	DMKIOS	DMKSPL	DMKSYM	DMKUNT	DMKVCA	DMKVSI	
DMKVIOMK	000013	DMKCFM	DMKCFP	DMKCPB	DMKDSP	DMKSSS	DMKVCN	DMKVDC	DMKVSI	DMKVSP			
DMKVIOXK	000002	DMKVSI											
DMKVMASH	000048	DMKATS	DMKCCW	DMKCDM	DMKCDS	DMKCPI	DMKDGD	DMKDSP	DMKPGS	DMKSYM	DMKUSO	DMKVCN	DMKVSP
DMKVMASW	000006	DMKCPU	DMKLOK										
DMKVMAS1	000007	DMKCFG	DMKCFH	DMKCPU									
DMKVMAS2	000001	DMKCFG											
DMKVMC	000001	DMKSYM											
DMKVMCEX	000002	DMKDSP											
DMKVMCFC	000004	DMKHVC	DMKMSG										
DMKVMCUA	000002	DMKCFP											
DMKVMI	000004	DMKCFG	DMKCPI	DMKRPA	DMKSYM								
DMKVSICI	000001	DMKMIA											
DMKVSICT	000002	DMKMIA	DMKSYM										
DMKVSICW	000002	DMKMIA	DMKSYM										
DMKVSIEX	000006	DMKHVC	DMKPRV	DMKSYM									

LABEL	COUNT	REFERENCES											
DMKVSIHD	000001	DMKMIA											
DMKVSIHI	000001	DMKMIA											
DMKVSISF	000001	DMKMIA											
DMKVSISI	000001	DMKMIA											
DMKVSITC	000001	DMKMIA											
DMKVSITI	000001	DMKMIA											
DMKVSIVS	000001	DMKPRV											
DMKVSP	000001	DMKVSQ											
DMKVSPCO	000011	DMKCFP	DMKCPS	DMKCSP	DMKCSQ	DMKSYM	DMKVDR						
DMKVSPCP	000001	DMKSYM											
DMKVSPCR	000009	DMKCFP	DMKCSQ	DMKDRD	DMKSYM	DMKVDR							
DMKVSPEX	000003	DMKSYM	DMKVSI										
DMKVSPRT	000017	DMKCDM	DMKRNH	DMKSYM	DMKTRC	DMKTRD							
DMKVSPTO	000003	DMKSYM	DMKVSI										
DMKVSPVP	000005	DMKQCN	DMKSYM										
DMKVSPWA	000004	DMKSYM	DMKUSO	DMKVSQ									
DMKVSQPD	000011	DMKSYM	DMKVSP										
DMKWRM	000002	DMKLD00E	DMKSYM										
DMKWRMST	000002	DMKCPI											
DMPABEND	000001	DMKDMP											
DMPFLAG	000001	DMKDMP											
DMPFPRS	000001	DMKDMP											
DMPGPRS	000001	DMKDMP											
DMPINREC	000002	DMKDMP											
DMPKEY	000002	DMKDMP											
DMPKYREC	000001	DMKDMP											
DMPLCORE	000001	DMKDMP											
DMPPGMAP	000003	DMKDMP											
DMPPRFRG	000001	DMKDMP											
DMPPROCA	000001	DMKDMP											
DMPSYSRV	000001	DMKDMP											
DMPTODCK	000001	DMKDMP											
DUMPSAVE	000010	DMKDMP	DMKMCT	DMKPRG	DMKPSA	DMKSVC							
ECBLOK	000069	DMKBLD	DMKCDB	DMKCDM	DMKCDS	DMKCFG	DMKCFH	DMKCFP	DMKCFS	DMKDSP	DMKEXT	DMKPRG	DMKPRV
		DMKSCH	DMKSVC	DMKTMR	DMKTRC	DMKTRD	DMKUSC	DMKVAT	DMKVMC				
ECSWBYT3	000001	DMKCCH											
ECSWLOG	000009	DMKCCH	DMKIOG										
EDIT	000036	DMKCFH	DMKCFM	DMKCFO	DMKCNS	DMKCPI	DMKGRF	DMKLNK	DMKMSW	DMKNLD	DMKNLE	DMKQCN	DMKRGA
		DMKRNH	DMKVCN										
EGPR0	000002	DMKIMG											
EGPR15	000007	DMKIMG											
EGPR8	000001	DMKIMG											
EMSINQSC	000002	DMKDSP											
EMSMASK	000007	DMKAPI	DMKCLK	DMKDSP	DMKLOK								
EMSPCLKC	000003	DMKCLK	DMKEXT										
EMSPEND	000022	DMKCLK	DMKDSP	DMKEXT	DMKMCT								

LABEL	COUNT	REFERENCES											
EMSPEXT	000004	DMKDSP	DMKEXT										
EMSPQUI	000007	DMKDSP	DMKEXT	DMKMCT									
EMSPSHD	000003	DMKEXT											
EMSPSYNC	000004	DMKCLK	DMKEXT										
EMSRCLKC	000001	DMKCLK											
EMSREC	000007	DMKCLK	DMKDSP	DMKEXT									
EMSREXT	000001	DMKDSP											
EMSRQUI	000001	DMKDSP											
EMSRSHD	000001	DMKEXT											
EMSRSYNC	000001	DMKCLK											
EQCHK	000009	DMKRSE											
ERRBLOK	000009	DMKIOE	DMKIOF										
ERRCCNT	000003	DMKIOE	DMKIOF										
ERRCCW	000006	DMKCKP	DMKIOE	DMKIOF	DMKLD00E								
ERRCODE	000002	DMKCQG											
ERRCONT	000001	DMKIOF											
ERRCORR	000003	DMKIOE	DMKIOF										
ERRHEADR	000002	DMKIOE											
ERRIOB	000012	DMKIOE	DMKIOF										
ERRIOER	000003	DMKIOE	DMKIOF										
ERRKEY	000004	DMKIOE	DMKIOF										
ERRMIOB	000004	DMKIOE	DMKIOF										
ERRMIOER	000002	DMKIOE	DMKIOF										
ERRMSG	000018	DMKCFH	DMKCFM	DMKCQP	DMKDDR	DMKDIR	DMKERM	DMKLNK	DMKNLD	DMKNLE	DMKPRG	DMKRNH	
ERRMSIZE	000001	DMKIOE											
ERROR	000079	DMKCPB	DMKDDR	DMKDIR	DMKEMA	DMKEMB	DMKEMC	DMKFMT	DMKMCD	DMKMNI	DMKMON	DMKNMT	DMKRND
		DMKRNH	DMKSSP	DMKVDA	DMKVDD								
ERRPARM	000004	DMKCDB	DMKCDM	DMKIOE	DMKIOF								
ERRSDR	000010	DMKIOE	DMKIOF										
ERRSIZE	000001	DMKIOE											
ERRVOLID	000003	DMKIOE	DMKIOF										
ERSAVE	000005	DMKDDR											
ESIDTB	000007	DMKLD00E											
ETX	000011	DMKGRF	DMKGRT	DMKGRW	DMKRGA	DMKRGB							
EUA	000012	DMKDDR	DMKDIR	DMKFMT	DMKGRF	DMKGRT	DMKGRW	DMKOPR	DMKRGA	DMKSSP			
EXDCCF	000001	DMKMCH											
EXDCNO	000001	DMKMCH											
EXDRESVD	000001	DMKMCH											
EXHAUST	000004	DMKMIA											
EXNPSW	000006	DMKCPI	DMKDSP	DMKLD00E	DMKSAV								
EXOPSW	000015	DMKAPI	DMKDSP	DMKEXT	DMKPSA								
EXTARCH	000007	DMKVAT											
EXTCCTRQ	000007	DMKBLD	DMKCDS	DMKCFP	DMKCFS	DMKTMR	DMKUSO						
EXTCOPY	000005	DMKVAT											
EXTCPTMR	000026	DMKCDS	DMKCFP	DMKSCH	DMKTMR								
EXTCPTRQ	000017	DMKBLD	DMKCFP	DMKCFS	DMKSCH	DMKTMR	DMKUSO						

LABEL	COUNT	REFERENCES											
EXTCR0	000024	DMKBLD	DMKCDB	DMKCDM	DMKCDS	DMKCFG	DMKCFP	DMKDSP	DMKPRV	DMKTRC	DMKVAT	DMKVMC	
EXTCR1	000003	DMKVAT											
EXTCR14	000002	DMKBLD	DMKCFP										
EXTCR15	000002	DMKBLD	DMKCFP										
EXTCR2	000006	DMKBLD	DMKCFP	DMKDSP									
EXTCR4	000003	DMKCFP	DMKDSP										
EXTCR7	000001	DMKDSP											
EXTCR8	000002	DMKEXT	DMKSVC										
EXTCR9	000002	DMKPRV	DMKTMR										
EXTMASK	000013	DMKAPI	DMKCFH	DMKCPI	DMKCPU	DMKEXT	DMKSVC	DMKTMR	DMKTRC				
EXTMODE	000035	DMKAPI	DMKCDS	DMKCFG	DMKCLK	DMKCPB	DMKCPI	DMKDMP	DMKDSP	DMKPRV	DMKSVC	DMKTRC	DMKTRD
		DMKVAT	DMKVER	DMKVMI									
EXTPERAD	000005	DMKDSP	DMKPRG	DMKPRV	DMKTMR								
EXTPERCD	000003	DMKDSP	DMKPRG										
EXTSEGLN	000004	DMKVAT											
EXTSHCR0	000010	DMKDSP	DMKPRV	DMKTMR	DMKTRC	DMKTRD	DMKVAT						
EXTSHCR1	000012	DMKDSP	DMKVAT										
EXTSHLEN	000008	DMKVAT											
EXTSHSEG	000005	DMKVAT											
EXTSIZE	000008	DMKBLD	DMKCFS	DMKDDR	DMKUSO								
EXTSTOLD	000003	DMKVAT											
EXTVSEGS	000005	DMKVAT											
FAILADD	000002	DMKCCH											
FAILCCW	000007	DMKBSC	DMKCCH										
FAILCSW	000012	DMKCCH											
FAILECSW	000004	DMKCCH											
FCBIO	000001	DMKTCS											
FCBNEXT	000001	DMKCSB											
FFS	000179	DMKAPI	DMKBLD	DMKCCH	DMKCCW	DMKCDB	DMKCDM	DMKCFC	DMKCFP	DMKCFS	DMKCKS	DMKCNS	DMKCPB
		DMKCPI	DMKCPS	DMKCQP	DMKCSP	DMKCSQ	DMKCST	DMKCSU	DMKCSV	DMKDAS	DMKDEF	DMKDGD	DMKDIA
		DMKDRD	DMKDSP	DMKEIG	DMKFRE	DMKHVC	DMKHVD	DMKIOF	DMKIOG	DMKIOS	DMKLNK	DMKLOG	DMKMCC
		DMKMCH	DMKMNI	DMKNES	DMKNLD	DMKOPR	DMKPGS	DMKPGT	DMKPRG	DMKPRV	DMKPTR	DMKRND	DMKRPA
		DMKSCH	DMKSCN	DMKSEV	DMKSIX	DMKSVC	DMKTAP	DMKTRA	DMKTRC	DMKTRD	DMKTRK	DMKUNT	DMKUSO
		DMKVCA	DMKVCH	DMKVDA	DMKVDC	DMKVDD	DMKVDR	DMKVDS	DMKVSP	DMKVSQ	DMKWRM		
FILE	000002	DMKSEP											
FINIS	000001	DMKTRK											
FLAG	000013	DMKCKP	DMKDIR	DMKFMT									
FLAG1	000002	DMKTRA	DMKVMI										
FLAG2	000003	DMKTRA	DMKVMI										
FNAME	000001	DMKEMA											
FPRLOG	000002	DMKDMP											
FREELOWE	000001	DMKNMT											
FREENUM	000008	DMKFRE											
FREEQ	000007	DMKPTR											
FREER0	000002	DMKFRE											
FREER1	000004	DMKFRE											

LABEL	COUNT	REFERENCES											
FREER14	000002	DMKFRE											
FREER15	000003	DMKFRE											
FREESAVE	000017	DMKFRE	DMKPTR	DMKVCA									
FRLKPROC	000006	DMKDSP	DMKFRE										
FSCBFN	000008	DMKRND											
FSIZE	000010	DMKLD00E											
FSTFMODE	000001	DMKNMT											
FSTFNAME	000001	DMKNMT											
FTRDIAL	000026	DMKBSC	DMKRGA	DMKRGB									
FTREXTSN	000007	DMKCCW	DMKDAS	DMKDSB	DMKIOE	DMKIOF	DMKRSE	DMKSSP					
FTROPRDR	000001	DMKACO											
FTRRPS	000008	DMKCPI	DMKDAS	DMKDSB	DMKIOS	DMKVDE							
FTRRSRL	000006	DMKCPI	DMKDAS	DMKDSB	DMKTAP	DMKVDS							
FTRTYP1	000003	DMKNLD	DMKNLE										
FTRUCS	000002	DMKCSB	DMKSSP										
FTRVIRT	000004	DMKCPI	DMKVDC	DMKVDE									
FTR2311B	000005	DMKDIR	DMKLNK	DMKSCN	DMKVER								
FTR2311T	000005	DMKDIR	DMKLNK	DMKSCN	DMKVER								
FTR35MB	000015	DMKCCW	DMKCKP	DMKCPI	DMKDAS	DMKDSB	DMKHVD	DMKUNT	DMKVDC	DMKVDE	DMKVER	DMKVSI	
FTR4WCGM	000001	DMKTCS											
FTR70MB	000012	DMKCPI	DMKDAS	DMKDSB	DMKPAG	DMKPGT	DMKSPL	DMKTDK	DMKVDE	DMKVSI			
FXDLOG	000004	DMKCCH	DMKDMP										
F0	000040	DMKCFG	DMKCFH	DMKCPI	DMKCQR	DMKFRE	DMKGRF	DMKIOG	DMKIOS	DMKJRL	DMKMCC	DMKMCD	DMKMIA
		DMKMON	DMKNLE	DMKPGS	DMKPTR	DMKRGA	DMKSCN	DMKSPL	DMKUNT	DMKVDC	DMKVSP	DMKWRM	
F1	000299	DMKACO	DMKALG	DMKAPI	DMKATS	DMKBLD	DMKBSC	DMKCCH	DMKCCW	DMKCDE	DMKCDM	DMKCDS	DMKCFC
		DMKCFD	DMKCFG	DMKCFH	DMKCFO	DMKCFP	DMKCFT	DMKCKP	DMKCKS	DMKCNS	DMKCPI	DMKCPU	DMKCPV
		DMKCQG	DMKCQH	DMKCQP	DMKCQR	DMKCQY	DMKCSE	DMKCSO	DMKCSP	DMKCSQ	DMKCST	DMKCSU	DMKCSV
		DMKCVT	DMKDAS	DMKDGD	DMKDIA	DMKDRD	DMKDSP	DMKENT	DMKFRE	DMKGRF	DMKHVC	DMKHVD	DMKIOE
		DMKIOG	DMKIOS	DMKLNK	DMKLOH	DMKLOK	DMKMCC	DMKMCD	DMKMIA	DMKMNI	DMKMON	DMKMSG	DMKNES
		DMKNLD	DMKNLE	DMKPAG	DMKPGS	DMKPGT	DMKPTR	DMKQCN	DMKRGA	DMKRGB	DMKRNH	DMKRPA	DMKRSE
		DMKSAV	DMKSCH	DMKSIX	DMKSNC	DMKSPL	DMKSSS	DMKSTK	DMKTAP	DMKTCS	DMKTHI	DMKTMR	DMKTRD
		DMKTRK	DMKUDR	DMKUNT	DMKVAT	DMKVCA	DMKVCH	DMKVCN	DMKVDC	DMKVDE	DMKVDR	DMKVER	DMKVMA
		DMKVMC	DMKVSI	DMKVSP	DMKVSQ								
F10	000026	DMKCCW	DMKCDB	DMKCDM	DMKCKS	DMKCNS	DMKCPI	DMKCVT	DMKDAS	DMKMCC	DMKMSW	DMKSCH	DMKVDC
		DMKVSI											
F15	000047	DMKATS	DMKBLD	DMKBSC	DMKCCW	DMKCDB	DMKCDS	DMKCFG	DMKCNS	DMKCPI	DMKCPU	DMKDGD	DMKIOF
		DMKLNK	DMKPGS	DMKPRV	DMKPSA	DMKRSE	DMKSCH	DMKTAP	DMKTRC	DMKTRD	DMKUNT	DMKUSO	DMKVER
		DMKVSI											
F16	000062	DMKATS	DMKBLD	DMKCCH	DMKCCW	DMKCDB	DMKCDS	DMKCFG	DMKCNS	DMKCPE	DMKCPU	DMKDAS	DMKDEF
		DMKDGD	DMKENT	DMKHVC	DMKHVD	DMKIOS	DMKISM	DMKPGS	DMKPRV	DMKPTR	DMKRNH	DMKSCH	DMKTAP
		DMKTRD	DMKUNT	DMKVAT	DMKVMA								
F2	000109	DMKCCW	DMKCDB	DMKCDM	DMKCDS	DMKCFC	DMKCFG	DMKCFH	DMKCFO	DMKCFS	DMKCFT	DMKCNS	DMKCPI
		DMKCPS	DMKCPV	DMKCQH	DMKCQP	DMKCQR	DMKCQY	DMKCSB	DMKCSO	DMKCSP	DMKCSQ	DMKCST	DMKCSU
		DMKCSV	DMKDAS	DMKDEF	DMKERM	DMKHVD	DMKIOS	DMKISM	DMKLNK	DMKMCC	DMKMCD	DMKMCH	DMKMIA
		DMKMNI	DMKMON	DMKMSG	DMKPAG	DMKPSA	DMKPTR	DMKRGA	DMKSAV	DMKSPL	DMKTAP	DMKTRC	DMKTRD
		DMKUDU	DMKVCA										

LABEL	COUNT	REFERENCES												
F20	000003	DMKHVD	DMKMSW	DMKRGA										
F24	000008	DMKCDB	DMKCDM	DMKCKS	DMKCSU	DMKHVD	DMKRSP	DMKVER						
F240	000017	DMKALG	DMKCCW	DMKCPI	DMKCVT	DMKDIA	DMKDIB	DMKPRV	DMKPSA	DMKTRD	DMKUDU	DMKUNT	DMKVCA	
		DMKVSI												
F255	000024	DMKBLD	DMKCCH	DMKCFT	DMKCKS	DMKCPU	DMKDRD	DMKDSP	DMKGRF	DMKIOE	DMKIOF	DMKIOG	DMKMCH	
		DMKMCT	DMKNES	DMKNET	DMKRGB	DMKTDK								
F256	000046	DMKATS	DMKCFG	DMKCFH	DMKCNS	DMKCPU	DMKDAS	DMKDRD	DMKEXT	DMKGRF	DMKHVC	DMKHVD	DMKNLD	
		DMKNLE	DMKRGB	DMKRNH	DMKSNC	DMKTCS	DMKTDK	DMKUDR	DMKVCN	DMKVER	DMKVMA	DMKWRM		
F3	000118	DMKCCW	DMKCDB	DMKCDM	DMKCFC	DMKCFD	DMKCFG	DMKCFH	DMKCFO	DMKCFS	DMKCKS	DMKCPB	DMKCPI	
		DMKCPS	DMKCPV	DMKCQG	DMKCQH	DMKCQP	DMKCSB	DMKCSO	DMKCSP	DMKCSQ	DMKCST	DMKCSU	DMKCSV	
		DMKDEF	DMKDGD	DMKDIA	DMKDSP	DMKGRF	DMKMCC	DMKMCD	DMKMCH	DMKMCT	DMKMIA	DMKMON	DMKMSG	
		DMKNES	DMKNET	DMKNLD	DMKNLE	DMKPAG	DMKPGT	DMKRGA	DMKRSE	DMKSAV	DMKSCH	DMKSPL	DMKTAP	
		DMKTCS	DMKTHI	DMKTRA	DMKTRD	DMKTRK	DMKVCN	DMKVDC	DMKVIO					
F4	000120	DMKACO	DMKATS	DMKBLD	DMKCCW	DMKCDB	DMKCDM	DMKCDS	DMKCFC	DMKCFG	DMKCFH	DMKCFO	DMKCKS	
		DMKCNS	DMKCPB	DMKCPI	DMKCPU	DMKCQY	DMKCSB	DMKCSO	DMKCSP	DMKCSU	DMKCSV	DMKCVT	DMKDEF	
		DMKDGD	DMKGRF	DMKGRT	DMKHVC	DMKHVD	DMKIOE	DMKIOF	DMKISM	DMKLNK	DMKMCC	DMKMCD	DMKMIA	
		DMKMON	DMKMSW	DMKNES	DMKNET	DMKNLE	DMKPAG	DMKPGT	DMKPRV	DMKRGA	DMKRGB	DMKRNH	DMKRPA	
		DMKRSE	DMKRSP	DMKSAV	DMKSPL	DMKTAP	DMKTHI	DMKTMR	DMKTRD	DMKUNT	DMKVAT	DMKVCN	DMKVDC	
		DMKVER	DMKVSI	DMKVSP										
F4095	000052	DMKACO	DMKBLD	DMKCCW	DMKCDB	DMKCFG	DMKCFT	DMKCPI	DMKCQR	DMKDAS	DMKDGD	DMKDIA	DMKHVC	
		DMKHVD	DMKLNK	DMKLOG	DMKMNI	DMKNES	DMKNET	DMKPSA	DMKQCN	DMKRGA	DMKRGB	DMKRNH	DMKRSP	
		DMKSVC	DMKTCS	DMKTMR	DMKVER	DMKVMA	DMKVSI	DMKVSP						
F4096	000081	DMKAPI	DMKATS	DMKCCW	DMKCDB	DMKCDM	DMKCFG	DMKCFH	DMKCPI	DMKCPU	DMKCPV	DMKDAS	DMKDGD	
		DMKDIR	DMKDRD	DMKDSP	DMKFRE	DMKHVC	DMKHVD	DMKMON	DMKNLD	DMKNLE	DMKPGS	DMKPTR	DMKRSP	
		DMKSNC	DMKSSP	DMKTCS	DMKUDR	DMKVSI	DMKVSP	DMKVSQ						
F5	000026	DMKCDS	DMKCFO	DMKCPI	DMKCSU	DMKDEF	DMKDGD	DMKGRF	DMKMCC	DMKMCD	DMKMIA	DMKMNI	DMKNLE	
		DMKPAG	DMKPRV	DMKRGA	DMKSCN	DMKTAP	DMKTMR							
F6	000028	DMKCDB	DMKCDM	DMKCDS	DMKCFC	DMKCFD	DMKCPB	DMKCQP	DMKCSO	DMKCSU	DMKCSV	DMKDGD	DMKMCH	
		DMKMSW	DMKPRV	DMKRGA	DMKTAP	DMKTRK	DMKVDC							
F60	000030	DMKACO	DMKCFC	DMKCQR	DMKCVT	DMKHVC	DMKHVD	DMKMCD	DMKNET	DMKPRV	DMKPSA	DMKRNH	DMKSVC	
		DMKTHI	DMKTMR	DMKTRC	DMKTRD									
F7	000054	DMKBLD	DMKBSC	DMKCCH	DMKCCW	DMKCFG	DMKCFO	DMKCFS	DMKCPI	DMKCPS	DMKCPU	DMKCQG	DMKCQP	
		DMKCSQ	DMKCSU	DMKIOE	DMKIOF	DMKIOG	DMKLNK	DMKLOG	DMKMCD	DMKPRV	DMKRSE	DMKSCN	DMKSEV	
		DMKSIX	DMKTMR	DMKTRM	DMKUNT	DMKVCN	DMKVDC	DMKVER	DMKVMC					
F8	000181	DMKACO	DMKATS	DMKBLD	DMKBSC	DMKCCH	DMKCCW	DMKCDS	DMKCFC	DMKCFG	DMKCFH	DMKCFO	DMKCFS	
		DMKCNS	DMKCPI	DMKCPS	DMKCPU	DMKCPV	DMKCQG	DMKCQH	DMKCQR	DMKCQY	DMKCSO	DMKCSP	DMKCSQ	
		DMKCSU	DMKCSV	DMKDAS	DMKDEF	DMKDGD	DMKDRD	DMKGRF	DMKHVD	DMKIOE	DMKIOF	DMKIOS	DMKISM	
		DMKLNK	DMKLOG	DMKMCC	DMKMCD	DMKMCH	DMKMNI	DMKMON	DMKMSW	DMKNET	DMKNLD	DMKNLE	DMKPAG	
		DMKPGS	DMKPSA	DMKRGA	DMKRNH	DMKRSE	DMKRSP	DMKSCH	DMKSEV	DMKSIX	DMKTAP	DMKTHI	DMKTMR	
		DMKTRA	DMKTRC	DMKTRD	DMKTRK	DMKUDR	DMKUSC	DMKVAT	DMKVCN	DMKVDA	DMKVDC	DMKVDD	DMKVDS	
		DMKVER	DMKVIO	DMKVMA	DMKVSI	DMKVSQ	DMKWRM							
F9	000009	DMKCCW	DMKCPI	DMKCPV	DMKMSW	DMKQCN	DMKUNT	DMKVDC						
GRAFDEV	000001	DMKDIA												
GRLOG	000001	DMKDMP												
GRTBLOK	000016	DMKGRF												
GRTCLRCP	000001	DMKGRF												

LABEL	COUNT	REFERENCES		
GRTCLRDS	000007	DMKGRF	DMKGRT	DMKGRW
GRTCLRIA	000001	DMKGRF		
GRTCLRL	000003	DMKGRF	DMKGRT	DMKGRW
GRTCPPCP	000001	DMKGRF		
GRTCPPDS	000006	DMKGRF	DMKGRT	DMKGRW
GRTCPPL	000003	DMKGRF	DMKGRT	DMKGRW
GRTCPRCP	000001	DMKGRF		
GRTCPRDS	000007	DMKGRF	DMKGRT	DMKGRW
GRTCPRL	000003	DMKGRF	DMKGRT	DMKGRW
GRTCRDCP	000001	DMKGRF		
GRTCRDDS	000013	DMKGRF	DMKGRT	DMKGRW
GRTCRDL	000006	DMKGRF	DMKGRT	DMKGRW
GRTEWRCP	000001	DMKGRF		
GRTHLDCP	000001	DMKGRF		
GRTHLDDS	000007	DMKGRF	DMKGRT	DMKGRW
GRTHLDL	000003	DMKGRF	DMKGRT	DMKGRW
GRTINHDS	000009	DMKGRF	DMKGRT	DMKGRW
GRTINHL	000006	DMKGRF	DMKGRT	DMKGRW
GRTMORCP	000001	DMKGRF		
GRTMORDS	000007	DMKGRF	DMKGRT	DMKGRW
GRTMORL	000003	DMKGRF	DMKGRT	DMKGRW
GRTMRDCP	000001	DMKGRF		
GRTMRDDS	000006	DMKGRF	DMKGRT	DMKGRW
GRTMRDL	000003	DMKGRF	DMKGRT	DMKGRW
GRTNACCP	000001	DMKGRF		
GRTNACDS	000007	DMKGRF	DMKGRT	DMKGRW
GRTNACL	000003	DMKGRF	DMKGRT	DMKGRW
GRTRMICP	000001	DMKGRF		
GRTRMIDS	000006	DMKGRF	DMKGRT	DMKGRW
GRTRMIL	000003	DMKGRF	DMKGRT	DMKGRW
GRTRUNCP	000001	DMKGRF		
GRTRUNDS	000013	DMKGRF	DMKGRT	DMKGRW
GRTRUNL	000006	DMKGRF	DMKGRT	DMKGRW
GRTRUNRL	000003	DMKGRF	DMKGRT	DMKGRW
GRTVMPCP	000001	DMKGRF		
GRTVMPDS	000006	DMKGRF	DMKGRT	DMKGRW
GRTVMPL	000003	DMKGRF	DMKGRT	DMKGRW
GRTVMRCP	000001	DMKGRF		
GRTVMRDS	000006	DMKGRF	DMKGRT	DMKGRW
GRTVMRL	000003	DMKGRF	DMKGRT	DMKGRW
GRTWINCP	000001	DMKGRF		
GRTWINDS	000007	DMKGRF	DMKGRT	DMKGRW
GRTWINL	000003	DMKGRF	DMKGRT	DMKGRW
GRTWRTCP	000001	DMKGRF		
GRTWRTDS	000004	DMKGRF	DMKGRT	DMKGRW
GRTWRTL	000003	DMKGRF	DMKGRT	DMKGRW

LABEL	COUNT	REFERENCES											
HALFPAGE	000002	DMKDMP	DMKVMA										
HARDSTOP	000002	DMKCPI	DMKDMP										
HEX	000001	DMKDIR											
HIOCCH	000006	DMKCCH	DMKSEV	DMKSIX									
HOLD	000004	DMKCPC	DMKUSO										
IC	000033	DMKDDR	DMKDIR	DMKFMT	DMKGRF	DMKGRT	DMKGRW	DMKOPR	DMKRGA	DMKRGB	DMKSSP		
IDA	000065	DMKCCW	DMKDAS	DMKDGD	DMKDIB	DMKISM	DMKTAP	DMKTCS	DMKTRD	DMKUNT	DMKVCA	DMKVCN	DMKVSI
		DMKVSP											
IDLEWAIT	000009	DMKAPI	DMKCPI	DMKDSP	DMKMON	DMKSCH							
IFCC	000058	DMKBSC	DMKCCH	DMKCNS	DMKCPI	DMKDAS	DMKDSB	DMKDSP	DMKEIG	DMKGRF	DMKHVC	DMKIOE	DMKIOS
		DMKMSW	DMKRSE	DMKRSP	DMKSEV	DMKSIX	DMKTAP	DMKUNT	DMKVIO	DMKVSI			
IGBLANE	000027	DMKEIG	DMKSEV	DMKSIX									
IGPRGFLG	000009	DMKCCH	DMKSEV	DMKSIX									
IGTERMSQ	000048	DMKCCH	DMKEIG	DMKSEV	DMKSIX								
IGVALIDB	000029	DMKCCH	DMKEIG	DMKSEV	DMKSIX								
IL	000039	DMKCNS	DMKDIB	DMKGIO	DMKHVC	DMKIOS	DMKNLD	DMKNLE	DMKPAG	DMKRNH	DMKRSP	DMKTAP	DMKTRK
		DMKVCA	DMKVCN	DMKVIO	DMKVMI	DMKVSP							
INHIBIT	000038	DMKACO	DMKCNS	DMKGRF	DMKLNK	DMKLOG	DMKQCN	DMKRGA	DMKRGB	DMKRNH	DMKVCN		
INPUT	000021	DMKDDR	DMKRND	DMKSSP									
INTERCCH	000005	DMKCCH	DMKEIG	DMKSEV									
INTEX	000010	DMKDSP	DMKEXT	DMKPSA									
INTEXF	000003	DMKDSP	DMKPSA										
INTKFLIN	000001	DMKPSA											
INTMASK	000005	DMKAPI	DMKCLK	DMKCPI									
INTMC	000001	DMKMCH											
INTPR	000019	DMKCKP	DMKPRG	DMKPRV	DMKVAT								
INTPRL	000009	DMKDMP	DMKDSP	DMKPRG	DMKPRV								
INTRC	000002	DMKMCH											
INTREQ	000039	DMKCNS	DMKCPI	DMKDDR	DMKDIB	DMKDMP	DMKGRF	DMKIOS	DMKMSW	DMKNLD	DMKNLE	DMKRNH	DMKRSE
		DMKSAV	DMKVCA	DMKVCN	DMKVIO	DMKVSP							
INTSVC	000005	DMKSVC											
INTSVCL	000013	DMKPRG	DMKSVC	DMKTRC									
INTTIO	000018	DMKCCH	DMKCKP	DMKDMP	DMKDSP	DMKIOS	DMKSAV	DMKVMI					
INUSE	000002	DMKPGS											
INVLD	000002	DMKCDB	DMKCDM										
IOBALTSK	000007	DMKCCW	DMKDAS	DMKTRK	DMKUNT	DMKVIO							
IOBBPNT	000017	DMKDSP	DMKIOS	DMKNLD	DMKPAG	DMKSSS							
IOBCAW	000201	DMKACO	DMKBSC	DMKCCW	DMKCFP	DMKCNS	DMKCPB	DMKCPI	DMKCSB	DMKDAS	DMKDGD	DMKDIB	DMKDSB
		DMKGIO	DMKGRF	DMKHVC	DMKIOG	DMKIOS	DMKISM	DMKMCC	DMKMNI	DMKMON	DMKNLD	DMKNLE	DMKPAG
		DMKRGA	DMKRGB	DMKRNH	DMKRSE	DMKRSP	DMKSEP	DMKSPL	DMKTAP	DMKTCS	DMKTRD	DMKTRK	DMKUDR
		DMKUNT	DMKVCA	DMKVDE	DMKVDR	DMKVIO	DMKVSI						
IOBCCH	000004	DMKCCH											
IOBCC1	000018	DMKCNS	DMKDGD	DMKDIB	DMKIOS	DMKNLD	DMKNLE	DMKRNH	DMKRSE	DMKRSP	DMKVCA	DMKVDE	
IOBCC2	000004	DMKIOG	DMKIOS	DMKVIO									
IOBCC3	000046	DMKCFP	DMKCNS	DMKCPS	DMKDGD	DMKDSB	DMKGIC	DMKIOS	DMKNLD	DMKNLE	DMKPAG	DMKRGA	DMKRNH
		DMKRSE	DMKTAP	DMKTRK	DMKUNT	DMKVCA	DMKVDC	DMKVDE	DMKVIO	DMKVSI			

LABEL	COUNT	REFERENCES											
IOBCLN	000004	DMKCCW	DMKGIO	DMKUNT	DMKVSI								
IOBCOPY	000006	DMKGRF											
IOBCP	000054	DMKACO	DMKCCH	DMKCPS	DMKCSO	DMKDAS	DMKDIA	DMKIOE	DMKIOS	DMKNLD	DMKNLE	DMKPAG	DMKRGA
		DMKRGB	DMKRNH	DMKRSP	DMKSPL	DMKTAP	DMKTDK	DMKTRK	DMKUDR	DMKVDC	DMKVDE		
IOBCSW	000331	DMKACO	DMKBSC	DMKCCH	DMKCFP	DMKCNS	DMKCPS	DMKCSB	DMKCSO	DMKCSP	DMKCSU	DMKCSV	DMKDAS
		DMKDDR	DMKDGD	DMKDIA	DMKDIB	DMKDIR	DMKDSB	DMKDSP	DMKGIO	DMKGRF	DMKHVC	DMKIOG	DMKIOS
		DMKMON	DMKNLD	DMKNLE	DMKPAG	DMKRGA	DMKRNH	DMKRSE	DMKRSP	DMKSEP	DMKSPL	DMKSSP	DMKTAP
		DMKTRC	DMKTRD	DMKTRK	DMKUNT	DMKVCA	DMKVDC	DMKVDD	DMKVDE	DMKVIO	DMKVSI		
IOBCYL	000026	DMKCCW	DMKDGD	DMKIOS	DMKMON	DMKPAG	DMKPGT	DMKSPL	DMKSSS	DMKTDK			
IOBERP	000029	DMKBSC	DMKCNS	DMKDAS	DMKGRF	DMKIOS	DMKRSE	DMKRSP	DMKTAP				
IOBFATAL	000096	DMKACO	DMKBSC	DMKCFP	DMKCNS	DMKCSB	DMKCSC	DMKDAS	DMKDGD	DMKDSB	DMKGIO	DMKGRF	DMKIOE
		DMKIOF	DMKIOG	DMKIOS	DMKMNI	DMKMON	DMKPAG	DMKRGA	DMKRSE	DMKRSP	DMKSEP	DMKSPL	DMKSSP
		DMKTAP	DMKTCS	DMKTRK	DMKUDR	DMKVDE	DMKVIO						
IOBFLAG	000178	DMKACO	DMKBSC	DMKCCH	DMKCCW	DMKCFP	DMKCNS	DMKCPS	DMKCSB	DMKCSO	DMKDAS	DMKDGD	DMKDIA
		DMKDIB	DMKDSP	DMKGIO	DMKGRF	DMKIOE	DMKIOS	DMKMNI	DMKMCN	DMKNLD	DMKNLE	DMKPAG	DMKRGA
		DMKRGB	DMKRNH	DMKRSE	DMKRSP	DMKSEP	DMKSPL	DMKTAP	DMKTCS	DMKTDK	DMKTRK	DMKUDR	DMKUNT
		DMKVCA	DMKVDC	DMKVDE	DMKVDR	DMKVIO	DMKVSI						
IOBFLT	000005	DMKIOS	DMKSSS										
IOBFPNT	000043	DMKDSP	DMKIOS	DMKNLD	DMKPAG	DMKPGT	DMKSSS	DMKSTK	DMKVSI				
IOBHIO	000023	DMKCCH	DMKCFP	DMKCPS	DMKIOS	DMKVIO	DMKVSI						
IOBHVC	000014	DMKCFP	DMKCPS	DMKDGD	DMKGIO	DMKIOE	DMKIOS	DMKTRK					
IOBIMSTK	000003	DMKCPS	DMKIOS										
IOBIOER	000141	DMKBSC	DMKCCH	DMKCFP	DMKCNS	DMKCPS	DMKDAS	DMKDGD	DMKDIA	DMKDIB	DMKDSB	DMKGIO	DMKGRF
		DMKIOE	DMKIOS	DMKMNI	DMKMON	DMKNLD	DMKNLE	DMKRGA	DMKRGB	DMKRNH	DMKRSE	DMKRSP	DMKTAP
		DMKTRK	DMKVCA	DMKVDC	DMKVDE	DMKVIO							
IOBIRA	000087	DMKACO	DMKCFP	DMKCNS	DMKCPB	DMKCPI	DMKCPS	DMKCSB	DMKCSO	DMKCSP	DMKCSU	DMKCSV	DMKDGD
		DMKDIA	DMKDSB	DMKDSP	DMKGIO	DMKGRF	DMKIOG	DMKIOS	DMKISM	DMKMNI	DMKNLD	DMKNLE	DMKPAG
		DMKRGA	DMKRGB	DMKRNH	DMKRSE	DMKRSP	DMKSEP	DMKSPL	DMKTAP	DMKTCS	DMKTRK	DMKUDR	DMKUNT
		DMKVCA	DMKVDC	DMKVDD	DMKVDE	DMKVDR	DMKVSI						
IOBLINK	000055	DMKACO	DMKCFP	DMKCNS	DMKCPS	DMKCSB	DMKCSC	DMKCSP	DMKCSU	DMKCSV	DMKDGD	DMKDIA	DMKDSB
		DMKGIO	DMKGRF	DMKIOG	DMKIOS	DMKNLD	DMKNLE	DMKPAG	DMKRGA	DMKRNH	DMKSEP	DMKSPL	DMKTAP
		DMKTCS	DMKUNT	DMKVCA	DMKVDD	DMKVIO	DMKVSI						
IOBLOK	000316	DMKACO	DMKBSC	DMKCCH	DMKCCW	DMKCFP	DMKCNS	DMKCPB	DMKCPI	DMKCPS	DMKCSB	DMKCSO	DMKCSP
		DMKCSU	DMKCSV	DMKDAS	DMKDGD	DMKDIA	DMKDIE	DMKDSB	DMKDSP	DMKGIO	DMKGRF	DMKHVC	DMKIOE
		DMKIOG	DMKIOS	DMKISM	DMKLOG	DMKMCC	DMKMNI	DMKMON	DMKMSW	DMKNLD	DMKNLE	DMKPAG	DMKPGT
		DMKRGA	DMKRGB	DMKRNH	DMKRSE	DMKRSP	DMKSEP	DMKSPL	DMKSSS	DMKSTK	DMKTAP	DMKTCS	DMKTDK
		DMKTRC	DMKTRD	DMKTRK	DMKUDR	DMKUNT	DMKUSC	DMKVCA	DMKVDC	DMKVDD	DMKVDE	DMKVDR	DMKVIO
		DMKVSI											
IOBMINI	000008	DMKIOS	DMKPAG										
IOBMISC	000135	DMKACO	DMKCCW	DMKCFP	DMKCNS	DMKCPB	DMKCPS	DMKCSO	DMKDGD	DMKDIA	DMKGIO	DMKGRF	DMKISM
		DMKMCC	DMKMNI	DMKMON	DMKNLD	DMKNLE	DMKPAG	DMKRGA	DMKRGB	DMKRNH	DMKRSE	DMKRSP	DMKSEP
		DMKTAP	DMKTCS	DMKTDK	DMKUDR	DMKUNT	DMKUSC	DMKVDC	DMKVDE	DMKVIO	DMKVSI		
IOBMISC2	000117	DMKACO	DMKCCW	DMKCFP	DMKCPS	DMKCSB	DMKDGD	DMKGIO	DMKGRF	DMKMNI	DMKMON	DMKNLD	DMKNLE
		DMKRGA	DMKRGB	DMKRNH	DMKRSE	DMKSEP	DMKSPL	DMKTCS	DMKTRK	DMKUDR	DMKVDC	DMKVDE	DMKVIO
IOBMSIZE	000002	DMKCPI											
IOBPAG	000008	DMKDSP	DMKIOS	DMKPAG									

LABEL	COUNT	REFERENCES											
IOBPATHF	000013	DMKCPS	DMKIOS										
IOBRADD	000074	DMKACO	DMKCCH	DMKCNS	DMKCPS	DMKCSB	DMKCSO	DMKDAS	DMKDIA	DMKDSB	DMKGRF	DMKHVC	DMKIOE
		DMKIOG	DMKIOS	DMKMSW	DMKNLD	DMKNLE	DMKPAG	DMKRGA	DMKRNH	DMKRSE	DMKRSP	DMKSPL	DMKSSS
		DMKTAP	DMKTRC	DMKTRD	DMKVDD	DMKVSI							
IOBRCAW	000078	DMKBSC	DMKDAS	DMKDIA	DMKDIB	DMKIOS	DMKNLD	DMKNLE	DMKRNH	DMKRSE	DMKRSP	DMKSSP	DMKTAP
		DMKTRK	DMKVSI										
IOBRCNT	000095	DMKBSC	DMKDAS	DMKGRF	DMKNLD	DMKNLE	DMKRGA	DMKRGB	DMKRNH	DMKRSE	DMKRSP	DMKTAP	DMKTRK
IOBREL	000005	DMKDAS	DMKDSB										
IOBRELCU	000016	DMKCCW	DMKIOS	DMKVDR	DMKVIO	DMKVSI							
IOBRES	000007	DMKCFP	DMKCNS	DMKIOS	DMKUNT	DMKVCA							
IOBRETRY	000003	DMKIOS											
IOBRSTRT	000066	DMKBSC	DMKCSB	DMKDAS	DMKDIB	DMKIOS	DMKNLD	DMKNLE	DMKRGA	DMKRGB	DMKRNH	DMKRSE	DMKRSP
		DMKSEP	DMKTAP	DMKTCS	DMKTRK	DMKVCA							
IOBRSV3	000003	DMKSSS											
IOBSENS	000004	DMKGRF											
IOBSIOF	000005	DMKIOS	DMKTRK	DMKVIO	DMKVSI								
IOBSIZE	000202	DMKACO	DMKCCW	DMKCFP	DMKCNS	DMKCPB	DMKCPI	DMKCPS	DMKCSB	DMKCSO	DMKCSP	DMKCSU	DMKCSV
		DMKDDR	DMKDGD	DMKDIA	DMKDIR	DMKDMP	DMKDSB	DMKGIO	DMKGRF	DMKHVC	DMKIOG	DMKIOS	DMKMCC
		DMKMNI	DMKMON	DMKNLD	DMKNLE	DMKPAG	DMKRGA	DMKRGB	DMKRNH	DMKRSE	DMKRSP	DMKSEP	DMKSPL
		DMKSSS	DMKTAP	DMKTCS	DMKTRK	DMKUDR	DMKUNT	DMKVCA	DMKVDC	DMKVDD	DMKVDE	DMKVDR	DMKVIO
		DMKVSI											
IOBSNSIO	000009	DMKIOS											
IOBSPEC	000098	DMKACO	DMKCCH	DMKCFP	DMKCNS	DMKCPS	DMKDSB	DMKGRF	DMKIOS	DMKNLD	DMKNLE	DMKRGA	DMKRGB
		DMKRNH	DMKRSE	DMKRSP	DMKTAP	DMKTRK	DMKVDC	DMKVDD	DMKVDE	DMKVIO	DMKVSI		
IOBSPEC2	000029	DMKCCW	DMKDAS	DMKDSB	DMKGIO	DMKIOS	DMKUNT	DMKVIO	DMKVSI				
IOBSPLT	000010	DMKIOS											
IOBSTAT	000222	DMKACO	DMKBSC	DMKCCW	DMKCFP	DMKCNS	DMKCPS	DMKCSB	DMKCSO	DMKDAS	DMKDDR	DMKDGD	DMKDIB
		DMKDIR	DMKDSB	DMKGIO	DMKGRF	DMKIOE	DMKIOG	DMKIOS	DMKMNI	DMKMON	DMKNLD	DMKNLE	DMKPAG
		DMKRGA	DMKRGB	DMKRNH	DMKRSE	DMKRSP	DMKSEP	DMKSPL	DMKSSS	DMKTAP	DMKTCS	DMKTRD	DMKTRK
		DMKUDR	DMKUNT	DMKVCA	DMKVDC	DMKVDE	DMKVIO	DMKVSI					
IOBTIO	000048	DMKCCH	DMKCFP	DMKCPS	DMKDSB	DMKIOS	DMKNLD	DMKNLE	DMKRSE	DMKTAP	DMKVDC	DMKVDE	DMKVIO
		DMKVSI											
IOBUC	000008	DMKIOS											
IOBUNREL	000002	DMKCCW	DMKUNT										
IOBUNSL	000022	DMKCFP	DMKCNS	DMKCPS	DMKGRF	DMKIOS	DMKNLD	DMKNLE	DMKRGA	DMKRNH	DMKRSP	DMKTAP	DMKVDD
		DMKVIO											
IOBUSER	000078	DMKACO	DMKCCH	DMKCFP	DMKCNS	DMKCPB	DMKCPI	DMKCPS	DMKCSB	DMKCSO	DMKCSP	DMKCSU	DMKCSV
		DMKDIA	DMKDIB	DMKDSB	DMKDSP	DMKGRF	DMKIOG	DMKIOS	DMKLCG	DMKMCC	DMKNLD	DMKNLE	DMKPAG
		DMKRGA	DMKRGB	DMKRNH	DMKRSE	DMKSPL	DMKSTK	DMKTAP	DMKTRK	DMKUDR	DMKUNT	DMKVCA	DMKVDC
		DMKVDD	DMKVDE	DMKVDR	DMKVSI								
IOBVADD	000021	DMKCFP	DMKCPS	DMKCSP	DMKCSU	DMKCSV	DMKDIA	DMKIOS	DMKSPL	DMKSSS	DMKTRC	DMKUNT	DMKVCA
		DMKVIO	DMKVSI										
IOBWRAP	000003	DMKCCW	DMKVIO										
IOELPNTR	000006	DMKCCH	DMKEIG	DMKIOG									
IOERACT	000006	DMKBSC	DMKDAS	DMKMSW	DMKRSE	DMKTAP							
IOERADR	000028	DMKDAS	DMKIOF	DMKMSW	DMKTAP	DMKTRK							

LABEL	COUNT	REFERENCES											
IOERALTR	000004	DMKDAS	DMKTRK										
IOERBLOK	000184	DMKBSC	DMKCCH	DMKCCW	DMKCFP	DMKCNS	DMKCPS	DMKDAS	DMKDGD	DMKDIA	DMKDIB	DMKDSB	DMKEIG
		DMKGIO	DMKGRF	DMKIOE	DMKIOF	DMKIOS	DMKMSW	DMKNLD	DMKNLE	DMKRGA	DMKRGB	DMKRNH	DMKRSE
		DMKRSP	DMKSEV	DMKSIX	DMKTAP	DMKTRK	DMKUNT	DMKVCA	DMKVDC	DMKVDE	DMKVIO	DMKVSI	
IOERBSR	000015	DMKTAP											
IOERB80	000002	DMKCCH											
IOERCAL	000003	DMKDAS											
IOERCAN	000019	DMKBSC	DMKDAS	DMKTAP									
IOERCCH	000004	DMKCCH											
IOERCCRA	000011	DMKBSC	DMKCCH	DMKDAS	DMKDSB	DMKIOE	DMKRSE	DMKRSP	DMKTAP				
IOERCCRL	000011	DMKBSC	DMKCCH	DMKDAS	DMKDSB	DMKIOE	DMKRSE	DMKRSP	DMKTAP				
IOERCCUA	000001	DMKCCH											
IOERCCW	000008	DMKDIB	DMKIOS	DMKVCA									
IOERCEMD	000014	DMKDAS	DMKIOE	DMKRSE	DMKTRK								
IOERCHID	000008	DMKCCH											
IOERCLN	000006	DMKTAP											
IOERCLOG	000001	DMKCCH											
IOERCNCL	000002	DMKMSW											
IOERCSW	000094	DMKBSC	DMKCCH	DMKDAS	DMKDIB	DMKDSB	DMKGIC	DMKGRF	DMKIOE	DMKIOF	DMKIOS	DMKMSW	DMKRSE
		DMKRSP	DMKTAP	DMKTRK	DMKVCA	DMKVIO							
IOERCYLR	000002	DMKUNT											
IOERDASD	000007	DMKDAS	DMKMSW										
IOERDATA	000250	DMKBSC	DMKCCW	DMKCNS	DMKCPS	DMKDAS	DMKDIB	DMKDSB	DMKGIO	DMKGRF	DMKIOE	DMKIOF	DMKIOS
		DMKMSW	DMKNLD	DMKNLE	DMKRNH	DMKRSE	DMKRSP	DMKTAP	DMKTRK	DMKUNT	DMKVCA	DMKVDC	DMKVDE
		DMKVIO											
IOERDEC	000004	DMKDAS	DMKMSW										
IOERDEPD	000003	DMKRSE	DMKRSP										
IOERDERD	000004	DMKRSE	DMKRSP										
IOERDW	000019	DMKBSC	DMKDAS	DMKDSB	DMKTAP	DMKTRK							
IOERECF	000003	DMKDAS											
IOERECSW	000004	DMKCCH	DMKRSE										
IOERERG	000004	DMKTAP											
IOERERP	000004	DMKRSE	DMKRSP										
IOERETN	000030	DMKNLD	DMKNLE	DMKPTR	DMKRPA	DMKSNC	DMKSVC	DMKUDR	DMKUDU	DMKVSP	DMKVSQ		
IOERETRY	000005	DMKDAS	DMKMSW	DMKRSE									
IOEREXT	000057	DMKBSC	DMKCCH	DMKCCW	DMKCFP	DMKCNS	DMKCPS	DMKDAS	DMKDGD	DMKDIA	DMKDSB	DMKGIO	DMKGRF
		DMKIOE	DMKIOF	DMKIOS	DMKNLD	DMKNLE	DMKRGA	DMKRGB	DMKRNH	DMKRSE	DMKRSP	DMKTAP	DMKTRK
		DMKVDC	DMKVDE	DMKVIO	DMKVSI								
IOERFLG1	000063	DMKDAS	DMKMSW	DMKRSE	DMKRSP	DMKTAP							
IOERFLG2	000060	DMKBSC	DMKDAS	DMKIOE	DMKRSE	DMKTAP	DMKTRK	DMKUNT					
IOERFLG3	000022	DMKBSC	DMKCNS	DMKDAS	DMKGRF	DMKIOF	DMKRSE	DMKTAP	DMKTRK				
IOERFSR	000011	DMKTAP											
IOERHA	000004	DMKDAS											
IOERIGN	000003	DMKMSW	DMKRSE										
IOERIGNR	000004	DMKDAS	DMKMSW	DMKTAP									
IOERIND3	000043	DMKBSC	DMKDAS	DMKMSW	DMKRSE	DMKTAP							

LABEL	COUNT	REFERENCES											
IOERIND4	000011	DMKDAS	DMKMSW	DMKTAP									
IOERINFO	000021	DMKBSC	DMKDAS	DMKMSW	DMKRSE	DMKTAP							
IOERLEN	000013	DMKCCW	DMKDSB	DMKIOF	DMKIOS	DMKMSW	DMKRSE	DMKUNT	DMKVCA				
IOERLG45	000001	DMKCCH											
IOERLOC	000034	DMKBSC	DMKDAS	DMKDSB	DMKTAP	DMKTRK							
IOERLOGL	000002	DMKCCH											
IOERMSG	000004	DMKDAS	DMKTAP										
IOERMSW	000013	DMKBSC	DMKDAS	DMKTAP									
IOERNUM	000072	DMKBSC	DMKCNS	DMKDAS	DMKGRF	DMKMSW	DMKRSE	DMKTAP					
IOERORA	000013	DMKTAP											
IOERPEND	000013	DMKDAS	DMKMSW	DMKRSE	DMKTAP								
IOERPNT	000022	DMKDAS	DMKIOE	DMKIOF	DMKRSE	DMKTRK							
IOERRBK	000012	DMKTAP											
IOERRDR0	000005	DMKDAS	DMKTRK										
IOERREAD	000015	DMKBSC	DMKCNS	DMKDAS	DMKGRF	DMKIOF	DMKRSE	DMKTAP					
IOERREW	000002	DMKTAP											
IOERSIZE	000075	DMKBSC	DMKCCH	DMKCCW	DMKCFP	DMKCNS	DMKCPS	DMKDAS	DMKDGD	DMKDIA	DMKDIB	DMKDSB	DMKGIO
		DMKGRF	DMKIOE	DMKIOF	DMKIOS	DMKMNI	DMKMON	DMKNLD	DMKNLE	DMKRGA	DMKRGB	DMKRNH	DMKRSE
		DMKRSP	DMKTAP	DMKTRK	DMKVCA	DMKVDC	DMKVDE	DMKVIO	DMKVSI				
IOERSNSZ	000014	DMKDAS	DMKDSB	DMKIOS	DMKRSE	DMKVDE							
IOERSTAT	000008	DMKDAS											
IOERSTRT	000003	DMKDAS	DMKMSW	DMKTAP									
IOERSUPP	000006	DMKTAP											
IOERS80	000002	DMKCCH											
IOERVLD	000003	DMKTAP											
IOERVOL1	000003	DMKDAS											
IOERVSER	000008	DMKDAS	DMKDSB	DMKIOF									
IOERWRK	000014	DMKTAP	DMKTRK										
IOERXERP	000003	DMKRSE											
IOERZCSW	000001	DMKCCH											
IOER2860	000002	DMKCCH											
IOER2870	000001	DMKCCH											
IOMASK	000006	DMKCFM	DMKDMP	DMKDSP	DMKLD00E	DMKSVC	DMKTRC						
IONPSW	000016	DMKCKP	DMKCPI	DMKDMP	DMKDSP	DMKFMT	DMKSAV	DMKSSP					
IONTWAIT	000008	DMKAPI	DMKCPI	DMKDSP	DMKMON	DMKSCH							
IOOLD	000019	DMKDDR	DMKDIR										
IOOPSW	000037	DMKCCH	DMKCKP	DMKDSP	DMKFMT	DMKIOS	DMKSAV	DMKSSP					
IOPSW	000001	DMKSSP											
IPLADDR	000001	DMKVMI											
IPLCCW1	000009	DMKCPI	DMKDMP	DMKVMI									
IPLPSW	000034	DMKCKP	DMKCPI	DMKDMP	DMKLD00E	DMKMIA	DMKMON	DMKVMI					
IPLREQ	000005	DMKNLD	DMKNLE	DMKRNH									
IPUADDR	000030	DMKAPI	DMKCFO	DMKCFS	DMKCPI	DMKCPU	DMKCQE	DMKCQY	DMKHVD	DMKIOG	DMKMCH	DMKMCT	
IPUADDRX	000045	DMKACO	DMKAPI	DMKCFO	DMKCLK	DMKCPI	DMKCPS	DMKCPU	DMKCQP	DMKCQY	DMKDSP	DMKEXT	DMKFRE
		DMKIOS	DMKMCH	DMKMCT	DMKPTR	DMKSCH							
IRMAND	000003	DMKCFO	DMKIOE										

LABEL	COUNT	REFERENCES										
IRMBIT1	000002	DMKCFO	DMKIOE									
IRMBIT2	000002	DMKCFO	DMKIOE									
IRMBLOK	000004	DMKCFO	DMKIOE									
IRMBYT1	000002	DMKCFO	DMKIOE									
IRMBYT2	000002	DMKCFO	DMKIOE									
IRMFLG	000006	DMKCFO	DMKIOE									
IRMLMT	000003	DMKCFO	DMKIOE									
IRMLMTCT	000003	DMKIOE										
IRMMAXCT	000005	DMKIOE										
IRMOR	000003	DMKCFO	DMKIOE									
IRMRLADD	000003	DMKCFO	DMKIOE									
IRMSIZE	000005	DMKCFO	DMKIOE									
IS	000001	DMKCQY										
JPSCBLOK	000014	DMKALG	DMKJRL	DMKLNK	DMKLOG							
JPSLNKAR	000003	DMKJRL										
JPSLNKDS	000001	DMKLNK										
JPSLNKMS	000003	DMKJRL										
JPSLNKU	000001	DMKJRL										
JPSLOGAR	000003	DMKJRL										
JPSLOGDS	000002	DMKALG	DMKLOG									
JPSLOGMS	000003	DMKJRL										
JPSLOGU	000001	DMKJRL										
JPSPCHN	000006	DMKJRL										
JRLSQOK	000002	DMKJRL										
KEEPSEGS	000014	DMKBLD	DMKCFG	DMKCFP	DMKPGS							
KEY	000003	DMKDMP										
KEYMASK	000005	DMKAPI	DMKCLK	DMKCPI								
LASTCYL	000003	DMKIOF	DMKIOG									
LASTLINE	000003	DMKDMP	DMKVDA									
LASTUSER	000045	DMKAPI	DMKATS	DMKBLD	DMKCDM	DMKDIA	DMKDSP	DMKFRE	DMKIOS	DMKLOK	DMKPAG	DMKPGS	DMKPTR
		DMKUSO											
LINE	000012	DMKDMP	DMKEMA	DMKFCB									
LINKJRL	000006	DMKJRL	DMKLNK										
LOCCNT	000001	DMKIMG											
LOCCT	000017	DMKLD00E											
LOCGRAF	000002	DMKACO											
LOCK	000248	DMKACO	DMKALG	DMKAPI	DMKATS	DMKBLD	DMKBSC	DMKCCH	DMKCCW	DMKCDB	DMKCDM	DMKCDS	DMKCFC
		DMKCFG	DMKCFH	DMKCFM	DMKCFO	DMKCFP	DMKCFS	DMKCFT	DMKCKP	DMKCKS	DMKCLK	DMKCNS	DMKCPB
		DMKCPE	DMKCPI	DMKCPS	DMKCPU	DMKCPV	DMKCSB	DMKCSO	DMKCSP	DMKCSQ	DMKCST	DMKCSU	DMKCSV
		DMKCVT	DMKDAS	DMKDEF	DMKDGD	DMKDIA	DMKDIB	DMKDMP	DMKDRD	DMKDSB	DMKDSP	DMKENT	DMKERM
		DMKEXT	DMKFCB	DMKFRE	DMKGIO	DMKGRF	DMKHVC	DMKHVD	DMKICE	DMKIOF	DMKIOG	DMKIOS	DMKJRL
		DMKLNK	DMKLOG	DMKLOH	DMKLOK	DMKMCC	DMKMCD	DMKMCH	DMKMCT	DMKMIA	DMKMID	DMKMNI	DMKMON
		DMKMSG	DMKMSW	DMKNES	DMKNET	DMKNLD	DMKNLE	DMKOPR	DMKPAG	DMKPGS	DMKPGT	DMKPRG	DMKPRV
		DMKPSA	DMKPTR	DMKQCN	DMKRGA	DMKRGB	DMKRNH	DMKRPA	DMKRSE	DMKRSP	DMKSAV	DMKSCH	DMKSEP
		DMKSNC	DMKSPL	DMKSSS	DMKSTK	DMKSVC	DMKSYM	DMKTAP	DMKTCS	DMKTDK	DMKTHI	DMKTMR	DMKTRA
		DMKTRC	DMKTRD	DMKTRK	DMKTRM	DMKUCB	DMKUCC	DMKUCS	DMKUNT	DMKUSO	DMKVAT	DMKVCA	DMKVCH

LABEL	COUNT	REFERENCES											
		DMKVCN	DMKVDA	DMKVDC	DMKVDD	DMKVDE	DMKVDR	DMKVDS	DMKVER	DMKVIO	DMKVMA	DMKVMC	DMKVMI
		DMKVSI	DMKVSP	DMKVSQ	DMKWRM								
LOCKBLOK	000004	DMKLOC											
LOCKLIST	000001	DMKPTR											
LOCKNAME	000002	DMKLOC											
LOCKNEXT	000004	DMKLOC											
LOCKQUE	000004	DMKLOC											
LOCKSAV	000056	DMKBLD	DMKDSP	DMKFRE	DMKIOS	DMKPRV	DMKPTR	DMKSCH	DMKTMR	DMKVMA	DMKVSI		
LOCKSIZE	000002	DMKLOC											
LOGDROP	000017	DMKCNS	DMKGRF	DMKQCN	DMKRGA	DMKRGB	DMKRNH	DMKUSO					
LOGHOLD	000016	DMKCNS	DMKDIA	DMKGRF	DMKQCN	DMKRGA	DMKRGB	DMKRNH	DMKUSO				
LOGONJRL	000007	DMKALG	DMKJRL	DMKLOG									
LOKSAVE	000015	DMKLOK											
LOKSAV2	000010	DMKLOK											
LPUADDR	000154	DMKAPI	DMKATS	DMKBLD	DMKCFG	DMKCFS	DMKCPI	DMKCPU	DMKCQR	DMKDSP	DMKEXT	DMKFRE	DMKIOS
		DMKLOC	DMKLOG	DMKLOK	DMKMCH	DMKMCT	DMKMNI	DMKPAG	DMKPRG	DMKPRV	DMKPSA	DMKPTR	DMKSCH
		DMKSTK	DMKSVC	DMKTMR	DMKVAT	DMKVMA	DMKVSI						
LPUADDRX	000019	DMKAPI	DMKCFS	DMKCPI	DMKCPU	DMKDSP	DMKFRE	DMKLOG	DMKMNI	DMKSTK			
MAINHIGH	000004	DMKIMG											
MASKLINK	000001	DMKLNK											
MASKLOG	000003	DMKLOG											
MCCPUID	000001	DMKMCH											
MCFXDLOG	000015	DMKMCH											
MCHAREA	000007	DMKCCH	DMKCFO	DMKCPU	DMKIOG	DMKMCH	DMKMCT						
MCHCPEX	000004	DMKCPU	DMKIOG	DMKMCH									
MCHDAMLN	000001	DMKIOG											
MCHEK	000009	DMKAPI	DMKCLK	DMKCPI	DMKDMP	DMKMCH	DMKSVC						
MCHFIX	000005	DMKCPU	DMKIOG	DMKMCH									
MCHFLAG0	000006	DMKMCH											
MCHFLAG1	000010	DMKMCH	DMKMCT										
MCHFLAG3	000009	DMKMCH											
MCHFLAG4	000002	DMKMCH											
MCHFLAG5	000001	DMKMCH											
MCHFLAG6	000001	DMKMCH											
MCHFLAG7	000027	DMKMCH	DMKMCT										
MCHFSAR	000006	DMKMCH											
MCHLEN	000002	DMKMCH											
MCHLEN1	000003	DMKCPU	DMKIOG	DMKMCH									
MCHMODEL	000024	DMKCCH	DMKCFO	DMKIOG	DMKMCH								
MCHPDAR1	000006	DMKMCH											
MCHPDAR6	000002	DMKMCH											
MCHPDAR7	000009	DMKMCH											
MCHPROCA	000001	DMKIOG											
MCHP1IDE	000001	DMKMCH											
MCHP1IKE	000002	DMKMCH											
MCHP1SDE	000001	DMKMCH											

LABEL	COUNT	REFERENCES						
MCHP1SKE	000002	DMKMCH						
MCHP6CBA	000002	DMKMCH						
MCHREC	000006	DMKCPU	DMKIOG	DMKMCH				
MCHRESEV	000004	DMKMCH						
MCH0HDWR	000002	DMKMCH						
MCH0QUIT	000001	DMKMCH						
MCH0SFTR	000001	DMKMCH						
MCH0TERM	000001	DMKMCH						
MCH0USAD	000001	DMKMCH						
MCH1BUFF	000001	DMKMCH						
MCH1COST	000002	DMKMCH						
MCH1GERR	000001	DMKMCH						
MCH1MAIN	000002	DMKMCH						
MCH1PROC	000002	DMKMCH						
MCH1TODC	000002	DMKMCH	DMKMCT					
MCH3DATA	000002	DMKMCH						
MCH3INTE	000002	DMKMCH						
MCH3PROT	000001	DMKMCH						
MCH3SOLD	000004	DMKMCH						
MCH4BURE	000001	DMKMCH						
MCH4REPA	000001	DMKMCH						
MCH5IFSA	000001	DMKMCH						
MCH7EXIT	000002	DMKMCH						
MCH7IOEM	000002	DMKMCH						
MCH7OPSW	000014	DMKMCH	DMKMCT					
MCH7PURG	000002	DMKMCH						
MCH7RSRE	000002	DMKMCH						
MCH7SMCR	000002	DMKMCH						
MCH7SUP	000003	DMKMCH						
MCH7SYST	000005	DMKMCH						
MCH7VEQR	000002	DMKMCH						
MCH7VRTM	000003	DMKMCH						
MCNPSW	000017	DMKCCH	DMKCKP	DMKCPI	DMKIOG	DMKMCH	DMKSAV	DMKSSP
MCOLDPW	000002	DMKMCH						
MCOPSW	000008	DMKMCH						
MCPROGID	000002	DMKMCH						
MCREC	000002	DMKMCH						
MCRECORD	000001	DMKMCH						
MCRECTYP	000001	DMKMCH						
MDRCUA1	000003	DMKVER						
MDRKEYN	000001	DMKVER						
MDRREC	000003	DMKIOF	DMKVER					
MDRSENS	000004	DMKVER						
MDRVOL	000001	DMKVER						
MFAMASK	000011	DMKAPI	DMKCLK	DMKCPI	DMKCPU	DMKDSP	DMKEXT	DMKLOK
MFASAVE	000002	DMKMCT						

LABEL	COUNT	REFERENCES							
MICBLOK	000021	DMKBLD	DMKCFS	DMKDSP	DMKLOG	DMKMCH	DMKPTR	DMKRPA	DMKTRA
MICCREG	000005	DMKCFS	DMKLOG						
MICEVMA	000002	DMKCFS	DMKLOG						
MICPEND	000002	DMKDSP							
MICRSEG	000003	DMKBLD	DMKCFS	DMKLOG					
MICSIZE	000006	DMKCFS	DMKLOG	DMKUSO					
MICVIP	000002	DMKDSP							
MICVPSW	000002	DMKCFS	DMKLOG						
MICVTMR	000014	DMKCFS	DMKLOG	DMKMCH	DMKPTR	DMKRPA	DMKTRA		
MICWORK	000002	DMKCFS	DMKLOG						
MIHCUA1	000002	DMKVER							
MIHKEYN	000001	DMKVER							
MIHREC	000002	DMKVER							
MIHSIZE	000001	DMKVER							
MIHVOL	000001	DMKVER							
MNBHDLEN	000002	DMKMCC	DMKMIA						
MNCHSAMP	000003	DMKENT							
MNCHSIZE	000005	DMKENT	DMKMCC	DMKMNI					
MNCLDAST	000002	DMKMNI	DMKMON						
MNCLINST	000004	DMKPRV							
MNCLPERF	000005	DMKMNI	DMKMON						
MNCLRESP	000006	DMKGRF	DMKQCN	DMKRGA					
MNCLSCH	000003	DMKSCH							
MNCLSEEK	000001	DMKIOS							
MNCLSYS	000001	DMKMON							
MNCLUSER	000002	DMKMNI	DMKMON						
MNCOAEL	000001	DMKSCH							
MNCOAQ	000001	DMKSCH							
MNCOBRD	000001	DMKQCN							
MNCOCYL	000001	DMKIOS							
MNCODA	000001	DMKMON							
MNCODAS	000001	DMKMON							
MNCODASH	000001	DMKMNI							
MNCODQ	000001	DMKSCH							
MNCOERD	000004	DMKGRF	DMKQCN	DMKRGA					
MNCOSIM	000004	DMKPRV							
MNCOSUS	000001	DMKMON							
MNCOSYS	000002	DMKMNI	DMKMON						
MNCOTH	000001	DMKMNI							
MNCOTT	000001	DMKMNI							
MNCOUSER	000002	DMKMNI	DMKMON						
MNCOWRIT	000001	DMKQCN							
MNCUBSY	000005	DMKENT							
MNDEVLEN	000005	DMKENT	DMKMNI	DMKMON					
MNDEVLST	000002	DMKENT							
MNDVBSY	000002	DMKENT							

LABEL	COUNT	REFERENCES
MNHCLASS	000001	DMKMON
MNHCODE	000001	DMKMON
MNHDR	000001	DMKMON
MNHDRLEN	000004	DMKMON
MNHRECSZ	000001	DMKMON
MNHTOD	000001	DMKMON
MN000	000002	DMKMON
MN000ATT	000001	DMKMON
MN000EXT	000001	DMKMON
MN000INT	000001	DMKMON
MN000ISD	000001	DMKMON
MN000LEN	000002	DMKMON
MN000PPA	000001	DMKMON
MN000PPC	000001	DMKMON
MN000PRB	000001	DMKMON
MN000PSI	000001	DMKMON
MN000Q1E	000002	DMKMON
MN000Q2E	000001	DMKMON
MN000WID	000001	DMKMON
MN000WIO	000001	DMKMON
MN000WPG	000001	DMKMON
MN001	000002	DMKMON
MN001LEN	000002	DMKMON
MN001NXR	000001	DMKMON
MN001PRB	000001	DMKMON
MN001WID	000001	DMKMON
MN001WIO	000001	DMKMON
MN001WPG	000001	DMKMON
MN097	000001	DMKMNI
MN097APL	000002	DMKMNI
MN097CPL	000002	DMKMNI
MN097CPU	000001	DMKMNI
MN097CR8	000001	DMKMNI
MN097DAT	000001	DMKMNI
MN097DPA	000001	DMKMNI
MN097FSS	000001	DMKMNI
MN097LEN	000001	DMKMNI
MN097LEV	000001	DMKMNI
MN097NUC	000001	DMKMNI
MN097TIM	000001	DMKMNI
MN097TTS	000001	DMKMNI
MN097UID	000001	DMKMNI
MN097VR	000001	DMKMNI
MN098	000001	DMKMNI
MN098LEN	000001	DMKMNI
MN098UID	000001	DMKMNI

LABEL	COUNT	REFERENCES
MN099	000001	DMKMON
MN099CNT	000001	DMKMON
MN099LEN	000001	DMKMON
MN099TOD	000001	DMKMON
MN10X	000001	DMKMON
MN10XADD	000002	DMKMON
MN10XLEN	000001	DMKMON
MN10XUID	000001	DMKMON
MN10YCNT	000001	DMKMON
MN10YIO	000001	DMKMON
MN10YLEN	000001	DMKMON
MN20X	000001	DMKMON
MN20XNPP	000001	DMKMON
MN20XPRC	000001	DMKMON
MN20XQNM	000008	DMKMON
MN20XQ1E	000001	DMKMON
MN20XQ1N	000001	DMKMON
MN20XQ2E	000001	DMKMON
MN20XQ2N	000001	DMKMON
MN20XSWS	000001	DMKMON
MN20XUID	000001	DMKMON
MN20XWSS	000001	DMKMON
MN20YTTI	000001	DMKMON
MN20YVTI	000001	DMKMON
MN202APR	000001	DMKMON
MN202CRD	000001	DMKMON
MN202IOC	000001	DMKMON
MN202LEN	000001	DMKMON
MN202LIN	000001	DMKMON
MN202LPR	000001	DMKMON
MN202PGR	000001	DMKMON
MN202PNC	000001	DMKMON
MN202PRI	000001	DMKMON
MN202PST	000001	DMKMON
MN202REF	000001	DMKMON
MN202RES	000001	DMKMON
MN203LEN	000001	DMKMON
MN204LEN	000001	DMKMON
MN204PRI	000001	DMKMON
MN4RSV1	000001	DMKMON
MN400	000001	DMKMON
MN400CRD	000001	DMKMON
MN400INT	000001	DMKMON
MN400IOC	000001	DMKMON
MN400LEN	000001	DMKMON
MN400LIN	000001	DMKMON

LABEL	COUNT	REFERENCES	
MN400LPR	000001	DMKMON	
MN400PDK	000001	DMKMON	
MN400PDR	000001	DMKMON	
MN400PGR	000001	DMKMON	
MN400PGW	000001	DMKMON	
MN400PNC	000001	DMKMON	
MN400PST	000001	DMKMON	
MN400QLV	000001	DMKMON	
MN400RES	000001	DMKMON	
MN400RST	000001	DMKMON	
MN400TTI	000001	DMKMON	
MN400UID	000001	DMKMON	
MN400UPR	000001	DMKMON	
MN400VTI	000001	DMKMON	
MN400WSS	000001	DMKMON	
MN500	000001	DMKMON	
MN500INS	000001	DMKMON	
MN500LEN	000001	DMKMON	
MN500OVH	000001	DMKMON	
MN500UID	000001	DMKMON	
MN500VAD	000002	DMKMON	
MN600ADD	000006	DMKMNI	DMKMON
MN600CNT	000002	DMKMNI	DMKMON
MN600DEV	000002	DMKMNI	DMKMON
MN600DLN	000004	DMKMNI	DMKMON
MN600HDR	000002	DMKMNI	DMKMON
MN600HLN	000004	DMKMNI	DMKMON
MN600MAX	000001	DMKMNI	
MN600NUM	000002	DMKMNI	DMKMON
MN600SER	000002	DMKMNI	DMKMON
MN600TY	000002	DMKMNI	DMKMON
MN602ADD	000003	DMKENT	
MN602CHB	000001	DMKENT	
MN602CHQ	000001	DMKENT	
MN602CUB	000001	DMKENT	
MN602CUQ	000001	DMKENT	
MN602DEV	000001	DMKENT	
MN602DLN	000002	DMKENT	DMKMON
MN602DVQ	000001	DMKENT	
MN602HDR	000001	DMKENT	
MN602HLN	000002	DMKENT	DMKMON
MN602SAM	000001	DMKENT	
MN700	000001	DMKMON	
MN700ADD	000001	DMKMON	
MN700CCY	000001	DMKMON	
MN700CYL	000001	DMKMON	

LABEL	COUNT	REFERENCES							
MN700DIR	000002	DMKMON							
MN700LEN	000001	DMKMON							
MN700QCH	000001	DMKMON							
MN700QCU	000001	DMKMON							
MN700QDV	000001	DMKMON							
MN700UID	000001	DMKMON							
MN802CLN	000001	DMKMON							
MN802CNT	000001	DMKMON							
MN802CTR	000001	DMKMON							
MN802DEV	000001	DMKMON							
MN802DLN	000002	DMKMON							
MN802NAU	000001	DMKMON							
MN802NPP	000001	DMKMON							
MN802NUM	000001	DMKMON							
MN802PGR	000001	DMKMON							
MN802PGW	000001	DMKMON							
MN802PRB	000001	DMKMON							
MN802WID	000001	DMKMON							
MN802WIO	000001	DMKMON							
MN802WPG	000001	DMKMON							
MODEL135	000004	DMKCCH	DMKIOG	DMKMCH					
MODEL138	000001	DMKIOG							
MODEL145	000004	DMKCCH	DMKIOG	DMKMCH					
MODEL148	000001	DMKIOG							
MODEL155	000004	DMKCCH	DMKIOG	DMKMCH					
MODEL158	000001	DMKIOG							
MODEL165	000005	DMKCCH	DMKIOG	DMKMCH					
MODEL168	000001	DMKIOG							
MOD3031	000002	DMKIOG	DMKMCH						
MOD3032	000001	DMKIOG							
MOD3033	000004	DMKCFO	DMKIOG	DMKMCH					
MOD4331	000005	DMKCCH	DMKIOG	DMKMCH					
MOD4341	000001	DMKIOG							
MONAIOB	000012	DMKCPS	DMKDMP	DMKMCC	DMKMIA	DMKMNI	DMKMON		
MONARDB	000007	DMKCPS	DMKDMP	DMKMCC	DMKMNI	DMKMON			
MONATRB	000006	DMKMCC	DMKMNI						
MONBUFAC	000010	DMKMCC	DMKMIA	DMKMON					
MONBUFAV	000007	DMKMCC	DMKMIA	DMKMON					
MONBUFIO	000007	DMKMIA	DMKMON						
MONBUF1	000012	DMKMCC	DMKMIA	DMKMNI	DMKMON				
MONBUF1V	000001	DMKMNI							
MONCHPTR	000012	DMKENT	DMKMCC	DMKMCD	DMKMNI	DMKMON			
MONCLASS	000014	DMKMNI	DMKMON	DMKPRG					
MONCLOCK	000004	DMKMON							
MONCODE	000018	DMKMNI	DMKMON	DMKPRG					
MONCOM	000025	DMKCPS	DMKDMP	DMKENT	DMKMCC	DMKMCD	DMKMIA	DMKMNI	DMKMON

LABEL	COUNT	REFERENCES							
MONCRSLT	000005	DMKMCC	DMKMON						
MONCURBF	000007	DMKMCC	DMKMIA	DMKMNI	DMKMON				
MONDAS	000014	DMKMIA							
MONDASA	000006	DMKMIA							
MONDASB	000006	DMKMIA							
MONDVLST	000012	DMKENT	DMKMCC	DMKMNI	DMKMON				
MONDVNUM	000013	DMKENT	DMKMCC	DMKMNI	DMKMON				
MONEX	000003	DMKMIA							
MONFLAG1	000034	DMKCPS	DMKMCC	DMKMCD	DMKMIA	DMKMNI	DMKMON		
MONFLAG2	000013	DMKDMP	DMKMIA	DMKMNI	DMKMON				
MONFLAG3	000048	DMKCPS	DMKDMP	DMKMCC	DMKMCD	DMKMIA	DMKMNI	DMKMON	
MONIOBF	000021	DMKCPS	DMKMCC	DMKMCD	DMKMIA	DMKMNI	DMKMON		
MONIOSLT	000004	DMKMCC	DMKMON						
MONLSTBK	000001	DMKMON							
MONNEXT	000015	DMKMCC	DMKMIA	DMKMNI	DMKMON				
MONREGS	000009	DMKMON	DMKPRG						
MONSACT	000004	DMKMON							
MONSAVE1	000001	DMKMON							
MONSAVE2	000001	DMKMON							
MONSFB	000006	DMKMIA	DMKMNI						
MONSIZE	000003	DMKMCC	DMKMNI						
MONSLMT	000005	DMKMCD	DMKMON						
MONSPLCT	000008	DMKMIA	DMKMNI	DMKMON					
MONSUSCK	000002	DMKMON							
MONSUSCT	000012	DMKMNI	DMKMON						
MONSYSVM	000003	DMKMON							
MONTIINT	000005	DMKMNI	DMKMON						
MONTINT	000001	DMKMCC							
MONUSER	000007	DMKCPS	DMKMCC	DMKMCD	DMKMIA	DMKMNI	DMKMON		
MONUTRB	000009	DMKENT	DMKMCC	DMKMCD	DMKMNI	DMKMON			
MON1BUF	000007	DMKMCC	DMKMIA	DMKMON					
MOUNT	000002	DMKSSS							
MPFEAT	000007	DMKATS	DMKCFG	DMKCPI	DMKCPU	DMKPGS			
MRDSIZE	000001	DMKVER							
MSGNUM	000003	DMKCNS	DMKGRF						
MSGTYPE	000001	DMKCKS							
MSSERR	000001	DMKSSS							
MSSFLAGS	000027	DMKLNK	DMKLOG	DMKSSS	DMKVDA				
MSSNEXT	000042	DMKCPB	DMKDGD	DMKDSB	DMKLNK	DMKLOG	DMKSSS	DMKVDA	DMKVSI
MSSPRES	000009	DMKCFP	DMKCPI	DMKDEF	DMKDGD	DMKLNK	DMKSSS	DMKUSO	DMKVSI
MSSRSRVD	000003	DMKSSS							
MSSSER	000009	DMKDSB	DMKSSS						
MSSSIZE	000014	DMKLNK	DMKLOG	DMKSSS	DMKVDA				
MSSTASK1	000016	DMKDGD	DMKDSB	DMKLNK	DMKSSS	DMKVDA	DMKVSI		
MSSTASK2	000014	DMKCPB	DMKDGD	DMKSSS					
MSSTASK3	000005	DMKLNK	DMKLOG	DMKVDA	DMKVSI				

LABEL	COUNT	REFERENCES											
MSSUSER	000006	DMKCPB	DMKDGD	DMKSSS	DMKVSI								
MSSVUA	000007	DMKDSB	DMKLOG	DMKSSS									
NCPNAME	000002	DMKNLD	DMKSNC										
NCPPAGCT	000002	DMKNLD	DMKSNC										
NCPPNT	000002	DMKNLD	DMKSNC										
NCPSTART	000002	DMKNLD	DMKSNC										
NCPTBL	000003	DMKNLD	DMKSNC										
NCPVOL	000004	DMKNLD	DMKSNC										
NEWEXT	000004	DMKDDR											
NEWPAGES	000016	DMKBLD	DMKCFG	DMKCFP	DMKCPI	DMKDEF	DMKLOG	DMKPTR					
NEWPSW	000001	DMKFMT											
NEWSEGS	000008	DMKBLD	DMKCFP	DMKCPI	DMKDEF	DMKLOG							
NICALRM	000005	DMKRGA	DMKRGB										
NICAPL	000010	DMKCFT	DMKCQR	DMKQCN	DMKRGA	DMKRGB							
NICATOF	000006	DMKCFT	DMKCQR	DMKRNH									
NICATRB	000006	DMKRGA	DMKRGB										
NICATTN	000008	DMKRNH											
NICBLOK	000039	DMKACO	DMKBLD	DMKCFT	DMKCKP	DMKCPI	DMKCQR	DMKDIA	DMKHVD	DMKLOG	DMKNES	DMKNET	DMKNLD
		DMKPSA	DMKQCN	DMKRGA	DMKRGB	DMKRNH	DMKWRM						
NICCARD	000005	DMKRGA											
NICCIBM	000006	DMKBLD	DMKDIA	DMKNES	DMKNET	DMKNLD	DMKRNH						
NICCORD	000007	DMKRGA	DMKRGB										
NICCPNA	000006	DMKRGA											
NICDED	000001	DMKRNH											
NICDIAG	000008	DMKRGA	DMKRGB										
NICDISA	000035	DMKCKP	DMKCPI	DMKDIA	DMKNES	DMKNET	DMKRGA	DMKRGB	DMKRNH	DMKWRM			
NICDISB	000014	DMKCKP	DMKNET	DMKRGA	DMKRGB	DMKRNH							
NICENAB	000027	DMKCKP	DMKDIA	DMKNES	DMKNET	DMKRGA	DMKRNH	DMKWRM					
NICEPAD	000009	DMKDIA	DMKNES	DMKNET	DMKNLD								
NICEPMD	000016	DMKDIA	DMKNES	DMKNET	DMKNLD	DMKRNH							
NICERLK	000004	DMKRNH											
NICFLAG	000096	DMKCFT	DMKCKP	DMKCQR	DMKDIA	DMKLOG	DMKNES	DMKNET	DMKNLD	DMKRGA	DMKRGB	DMKRNH	DMKWRM
NICFMT	000005	DMKRGA	DMKRGB										
NICGRAF	000003	DMKBLD	DMKHVD	DMKNET									
NICHOLD	000007	DMKRGA	DMKRGB										
NICLBSC	000002	DMKNES	DMKNET										
NICLGRP	000007	DMKCKP	DMKNET	DMKRGA	DMKWRM								
NICLINE	000013	DMKCKP	DMKDIA	DMKNES	DMKNET	DMKRNH							
NICLLEN	000007	DMKBLD	DMKCFT	DMKCQR	DMKHVD	DMKQCN							
NICLTRC	000013	DMKDIA	DMKNES	DMKRNH									
NICMORE	000007	DMKRGA	DMKRGB										
NICMTA	000002	DMKRNH											
NICNAME	000018	DMKBLD	DMKCPI	DMKDIA	DMKNET	DMKNLD	DMKPSA	DMKRGA	DMKRNH				
NICNTRL	000027	DMKRGA	DMKRGB	DMKRNH									
NICOPRDR	000001	DMKACO											
NICPOLL	000006	DMKRGA	DMKRGB										

LABEL	COUNT	REFERENCES											
NICPROCN	000006	DMKRGA	DMKRGB										
NICPSUP	000011	DMKCFT	DMKCQR	DMKLOG	DMKNES	DMKNLD	DMKRNH						
NICQPNT	000062	DMKDIA	DMKNES	DMKRGA	DMKRGB	DMKRNH							
NICRCNT	000012	DMKRNH											
NICREAD	000008	DMKRGA	DMKRGB										
NICRSPL	000007	DMKNET	DMKRGA										
NICRUNN	000011	DMKRGA	DMKRGB										
NICSELT	000005	DMKRGA	DMKRGB										
NICSESN	000012	DMKDIA	DMKNES	DMKNET	DMKRNH								
NICSIO	000006	DMKRGA	DMKRGB										
NICSIZE	000064	DMKACO	DMKCFT	DMKCKP	DMKCPI	DMKCPS	DMKCQR	DMKDIA	DMKHVD	DMKLOG	DMKNES	DMKNET	DMKNLD
		DMKPSA	DMKQCN	DMKRGA	DMKRGB	DMKRNH	DMKVDS	DMKWRM					
NICSTAT	000117	DMKCKP	DMKCPI	DMKDIA	DMKNES	DMKNET	DMKNLD	DMKRGA	DMKRGE	DMKRNH	DMKWRM		
NICSWEP	000007	DMKDIA	DMKNES	DMKNLD									
NICTABF	000003	DMKRGA											
NICTELE	000010	DMKDIA	DMKNET	DMKRNH									
NICTERM	000016	DMKBLD	DMKCKP	DMKNET	DMKNLD	DMKRGA	DMKRNH	DMKWRM					
NICTEXT	000013	DMKCFT	DMKCQR	DMKRGA	DMKRGB								
NICTMCD	000028	DMKCFT	DMKCQR	DMKHVD	DMKQCN	DMKRGA	DMKRGB						
NICTRQ	000006	DMKRGA	DMKRGB										
NICTYPE	000062	DMKACO	DMKBLD	DMKCKP	DMKDIA	DMKHVD	DMKNES	DMKNET	DMKNLD	DMKRGA	DMKRNH	DMKWRM	
NICUSER	000046	DMKBLD	DMKDIA	DMKLOG	DMKNES	DMKNET	DMKNLD	DMKPSA	DMKRGA	DMKRGB	DMKRNH		
NIC3275	000002	DMKRGA											
NOADD	000002	DMKCPS	DMKUDR										
NOAUTO	000031	DMKCFM	DMKCFO	DMKCNS	DMKCPI	DMKNLD	DMKNLE	DMKOPR	DMKQCN	DMKRNH	DMKVCN		
NOMODEL	000001	DMKIOG											
NOP	000007	DMKCCW	DMKLD00E	DMKTAP									
NORET	000445	DMKACO	DMKBLD	DMKCCH	DMKCDB	DMKCDM	DMKCDS	DMKCFC	DMKCFD	DMKCFH	DMKCFM	DMKCFO	DMKCFS
		DMKCPB	DMKCPI	DMKCPS	DMKCPU	DMKCPV	DMKCQG	DMKCQH	DMKCQP	DMKCQR	DMKCQV	DMKCSB	DMKCSO
		DMKCSU	DMKCSV	DMKDAS	DMKDEF	DMKDIA	DMKDSP	DMKERM	DMKGRF	DMKJRL	DMKLNK	DMKLOH	DMKMCC
		DMKMCD	DMKMCH	DMKMCT	DMKMIA	DMKMID	DMKMNI	DMKMSG	DMKMSW	DMKNES	DMKNET	DMKNLD	DMKNLE
		DMKPGT	DMKPRG	DMKPSA	DMKPTR	DMKQCN	DMKRGA	DMKRNH	DMKRSP	DMKSPL	DMKSVC	DMKTHI	DMKTRA
		DMKTRC	DMKTRD	DMKUDR	DMKUSO	DMKVCA	DMKVCH	DMKVCN	DMKVDA	DMKVDD	DMKVDR	DMKVER	
NOTEXT	000006	DMKGRF	DMKNMT	DMKRGA									
NOTIME	000061	DMKCFM	DMKCLK	DMKCPI	DMKGRF	DMKMSG	DMKMSW	DMKQCN	DMKRGA	DMKVCN			
NOTRESP	000018	DMKCFH	DMKCSV	DMKGRF	DMKLNK	DMKMSG	DMKNLD	DMKQCN	DMKRGA	DMKSPL	DMKVDA		
NPRCNT	000009	DMKCKS	DMKCSO	DMKHVD	DMKWRM								
NPRNAME	000006	DMKCKS	DMKCSO	DMKHVD	DMKTCS	DMKWRM							
NPRPAGCT	000002	DMKHVD											
NPRPNT	000010	DMKCKS	DMKCSO	DMKHVD	DMKTCS	DMKWRM							
NPRSTART	000002	DMKHVD	DMKTCS										
NPRTBL	000006	DMKCKS	DMKCSO	DMKHVD	DMKTCS	DMKWRM							
NPRVOL	000002	DMKHVD	DMKTCS										
NUCON	000002	DMKIMG	DMKNMT										
NUM	000004	DMKLD00E	DMKUCB	DMKUCC	DMKUCS								
OBRCORL	000001	DMKIOF											

LABEL	COUNT	REFERENCES											
OBRCPIDN	000002	DMKIOF	DMKVER										
OBRCSWN	000001	DMKIOF											
OBRCUA	000003	DMKVER											
OBRCUAIN	000004	DMKIOF	DMKVER										
OBRCUAPR	000004	DMKIOF	DMKVER										
OBRDDCNT	000005	DMKIOF											
OBRDEVSH	000011	DMKIOC											
OBRDEVTN	000011	DMKIOC											
OBREOD	000001	DMKIOF											
OBRFCCWN	000001	DMKIOF											
OBRHAN	000003	DMKIOF	DMKVER										
OBRHSIZE	000001	DMKVER											
OBRIORTY	000001	DMKIOF											
OBRKEYN	000010	DMKIOF	DMKVER										
OBRLSIZE	000001	DMKVER											
OBRLSKN	000003	DMKIOF	DMKVER										
OBRPGMN	000002	DMKIOF	DMKVER										
OBRRECN	000008	DMKIOC	DMKIOF	DMKVER									
OBRSDRCT	000006	DMKIOF											
OBRSDRSH	000001	DMKIOF											
OBRSENSN	000007	DMKVER											
OBRSHOBR	000008	DMKIOC	DMKIOF										
OBRSNSCT	000001	DMKIOF											
OBRSSDR1	000001	DMKIOF											
OBRSSIZE	000001	DMKVER											
OBRSWSN	000019	DMKIOC	DMKIOF	DMKVER									
OBRTEMP	000002	DMKIOF											
OBRVOLN	000003	DMKIOF	DMKVER										
OBR2SIZE	000001	DMKVER											
OBR3SIZE	000001	DMKVER											
OBR33SNS	000013	DMKIOF	DMKVER										
OFF	000006	DMKDSP	DMKLOK	DMKMCH	DMKMCT								
OLDVMSEG	000014	DMKBLD	DMKCFG	DMKCFP	DMKPGS								
ON	000007	DMKCPS	DMKRND										
OPERATOR	000125	DMKCCH	DMKCLK	DMKCPI	DMKCSO	DMKDAS	DMKDIA	DMKDSB	DMKERM	DMKLOH	DMKMCH	DMKMCT	DMKMIA
		DMKMSW	DMKNLD	DMKNLE	DMKPGT	DMKQCN	DMKRNH	DMKRSP	DMKUDR	DMKUSO	DMKVCH	DMKVDA	DMKVDD
		DMKVDR	DMKVER	DMKWRM									
OPNSFB	000010	DMKCKS	DMKMIA	DMKVSP									
OUTPUT	000002	DMKDDR	DMKLD00E										
OWNDLIST	000025	DMKATS	DMKCKP	DMKCKS	DMKCPI	DMKCPU	DMKDRD	DMKPAG	DMKPGS	DMKPGT	DMKPTR	DMKSPL	DMKUDR
		DMKVDA	DMKVDC	DMKWRM									
OWNDPREF	000002	DMKCPI	DMKVDA										
OWNDRDEV	000019	DMKATS	DMKCKP	DMKCKS	DMKCPI	DMKCPU	DMKDRD	DMKPAG	DMKPGS	DMKPGT	DMKPTR	DMKSPL	DMKUDR
		DMKVDC	DMKWRM										
OWNDVSER	000008	DMKCKS	DMKCPI	DMKUDR	DMKVDA								
PACK	000003	DMKCVT	DMKDDR	DMKDIR									

LABEL	COUNT	REFERENCES											
PAGACT	000035	DMKATS	DMKBLD	DMKCFG	DMKPGS	DMKPTR	DMKVMA						
PAGBMP	000035	DMKATS	DMKBLD	DMKCFG	DMKCPU	DMKPGS	DMKPTR						
PAGCORE	000089	DMKATS	DMKBLD	DMKCDS	DMKCFG	DMKCPI	DMKCPU	DMKMCH	DMKPGS	DMKPTR	DMKRPA	DMKVMA	
PAGECUR	000004	DMKCPU	DMKMCC	DMKMNI									
PAGELOAD	000004	DMKPAG	DMKSCH										
PAGEND	000009	DMKMCC	DMKMIA	DMKMNI	DMKMON								
PAGENXT	000001	DMKMNI											
PAGERATE	000003	DMKCPU	DMKPAG										
PAGEWAIT	000012	DMKAPI	DMKCPI	DMKCQR	DMKDSP	DMKMON	DMKPAG	DMKSCH					
PAGE4K	000008	DMKAPI	DMKCLK	DMKCPI									
PAGINVAL	000023	DMKATS	DMKCDS	DMKCPU	DMKMCH	DMKPTR	DMKRPA	DMKUDR	DMKVMA				
PAGREF	000010	DMKPGS	DMKPTR	DMKRPA									
PAGSHR	000015	DMKATS	DMKCFG	DMKPGS	DMKPTR								
PAGSTMP	000013	DMKATS	DMKBLD	DMKPGS	DMKPTR								
PAGSWP	000006	DMKATS	DMKBLD	DMKCFG	DMKVAT								
PAGTABLE	000031	DMKATS	DMKBLD	DMKCFG	DMKCPU	DMKPGS	DMKPTR	DMKVAT	DMKVMA				
PAGTONLY	000010	DMKBLD	DMKPGS	DMKPTR	DMKUSO								
PAGTOT	000013	DMKATS	DMKCFG	DMKCPU	DMKPGS								
PAGTSWP	000024	DMKATS	DMKCFG	DMKCPU	DMKPGS	DMKPTR							
PCHCHN	000005	DMKCKS	DMKSPL	DMKWRM									
PCI	000031	DMKDSP	DMKHVC	DMKIOS	DMKPAG	DMKRNH	DMKRSE	DMKTRK	DMKVCA	DMKVCN	DMKVIO	DMKVSI	DMKVSP
PCIF	000006	DMKCCW	DMKDGD	DMKVCA	DMKVCN	DMKVSP							
PERADD	000004	DMKDSP	DMKPRG										
PERCODE	000004	DMKDSP	DMKPRG										
PERFCL	000008	DMKMCC	DMKMIA	DMKMNI	DMKMON								
PERGPRS	000005	DMKPRV											
PERMODE	000003	DMKDSP	DMKTRC										
PERSALT	000008	DMKPRV	DMKTMR										
PGADDR	000002	DMKDSP	DMKVAT										
PGBLOK	000003	DMKCFP	DMKDSP	DMKVAT									
PGBSIZE	000003	DMKCFP	DMKDSP	DMKVAT									
PGPNT	000003	DMKCFP	DMKDSP	DMKVAT									
PGREAD	000009	DMKAPI	DMKCPI	DMKMIA	DMKMON	DMKPTR							
PGWAITIM	000004	DMKCPU	DMKDSP										
PGWRITE	000005	DMKMIA	DMKMON	DMKPTR									
PLIST	000001	DMKLD00E											
POFFLINE	000011	DMKAPI	DMKCPU	DMKMCT									
POINTER	000003	DMKWRM											
POINTERS	000003	DMKDIR											
PREFIXA	000089	DMKAPI	DMKATS	DMKBLD	DMKCDB	DMKCDM	DMKCDS	DMKCNS	DMKCPI	DMKCPU	DMKDSP	DMKEXT	DMKFRE
		DMKGRF	DMKHVD	DMKIOS	DMKLOK	DMKMCC	DMKMCH	DMKMCT	DMKMIA	DMKMNI	DMKMON	DMKPAG	DMKPRG
		DMKPSA	DMKPTR	DMKRGA	DMKRNH	DMKSCH	DMKSVC	DMKVIO	DMKVSI				
PREFIXB	000082	DMKAPI	DMKBLD	DMKCDB	DMKCDM	DMKCDS	DMKCFS	DMKCPI	DMKCPU	DMKCPV	DMKCQR	DMKCQY	DMKDSP
		DMKEXT	DMKHVD	DMKIOS	DMKLOG	DMKLOK	DMKMCC	DMKMCD	DMKMCH	DMKMCT	DMKMIA	DMKMID	DMKMNI
		DMKMON	DMKPAG	DMKPGS	DMKPSA	DMKPTR	DMKRPA	DMKSCH	DMKTHI	DMKVAT			
PRGC	000022	DMKBSC	DMKCNS	DMKDAS	DMKDIB	DMKGRF	DMKHVC	DMKIOS	DMKRNH	DMKRSE	DMKTAP	DMKUNT	DMKVCA

LABEL	COUNT	REFERENCES											
		DMKVCN	DMKVSP										
PRINTER1	000001	DMKDDR											
PRINTER2	000001	DMKDDR											
PRIORITY	000032	DMKACO	DMKCNS	DMKCPS	DMKCQR	DMKDIA	DMKGRF	DMKMCT	DMKMSG	DMKQCN	DMKRGB	DMKRNH	DMKUSO
		DMKVCN											
PRNPSW	000036	DMKAPI	DMKCKP	DMKCPI	DMKDMP	DMKDSP	DMKPRG	DMKSAV	DMKSSP				
PROBMODE	000012	DMKDSP	DMKMCH	DMKPRG	DMKPRV	DMKSVC							
PROBSTRT	000015	DMKCFP	DMKDSP	DMKSCH	DMKTMR								
PROBTIME	000015	DMKAPI	DMKCFP	DMKCPI	DMKDSP	DMKMON	DMKSCH	DMKTMR					
PROCIO	000157	DMKACO	DMKAPI	DMKATS	DMKBSC	DMKCDB	DMKCDM	DMKCDS	DMKCFG	DMKCFO	DMKCFP	DMKCFT	DMKCKS
		DMKCLK	DMKCNS	DMKCPI	DMKCPS	DMKCPU	DMKCPV	DMKCQP	DMKCQR	DMKCSO	DMKDAS	DMKDIA	DMKDSB
		DMKDSP	DMKENT	DMKFRE	DMKGRF	DMKHVD	DMKIOE	DMKIOG	DMKIOS	DMKLOG	DMKMCT	DMKMNI	DMKMON
		DMKNES	DMKNET	DMKNLD	DMKNLE	DMKPAG	DMKPGS	DMKPGT	DMKPSA	DMKPTR	DMKQCN	DMKRGA	DMKRGB
		DMKRNH	DMKRSE	DMKRSP	DMKSCH	DMKSEP	DMKSPL	DMKTAP	DMKTHI	DMKTRK	DMKTRM	DMKVCH	DMKVDA
		DMKVDD	DMKVDE	DMKVDR	DMKVDS	DMKVMA	DMKVSI						
PROCSCHK	000003	DMKCLK											
PROPSW	000045	DMKCKP	DMKCPI	DMKDMP	DMKDSP	DMKFMT	DMKMON	DMKPRG	DMKPRV				
PRTC	000019	DMKBSC	DMKCNS	DMKDAS	DMKDIB	DMKGRF	DMKHVC	DMKIOS	DMKRNH	DMKRSE	DMKTAP	DMKUNT	DMKVCA
		DMKVCN	DMKVSP										
PRTCHN	000007	DMKCKS	DMKSPL	DMKVSP	DMKWRM								
PSA	000379	DMKACO	DMKALG	DMKAPI	DMKATS	DMKBLD	DMKBSC	DMKCCH	DMKCCW	DMKCDB	DMKCDM	DMKCDS	DMKCFC
		DMKCFD	DMKCFG	DMKCFH	DMKCFM	DMKCFO	DMKCFP	DMKCFS	DMKCFT	DMKCKP	DMKCKS	DMKCLK	DMKCNS
		DMKCPB	DMKCPI	DMKCPS	DMKCPU	DMKCPV	DMKCQG	DMKCQH	DMKCQP	DMKCQR	DMKCQY	DMKCSB	DMKCSO
		DMKCSP	DMKCSQ	DMKCST	DMKCSU	DMKCSV	DMKCVT	DMKDAS	DMKDEF	DMKDGD	DMKDIA	DMKDIB	DMKDMP
		DMKDRD	DMKDSB	DMKDSP	DMKEIG	DMKENT	DMKERM	DMKEXT	DMKFMT	DMKFRE	DMKGIO	DMKGRF	DMKGRT
		DMKHVC	DMKHVD	DMKIOC	DMKIOE	DMKIOF	DMKIOG	DMKIOS	DMKISM	DMKJRL	DMKLNK	DMKLOC	DMKLOG
		DMKLOH	DMKLOK	DMKMCC	DMKMCD	DMKMCH	DMKMCT	DMKMIA	DMKMID	DMKMNI	DMKMON	DMKMSG	DMKMSW
		DMKNES	DMKNET	DMKNLD	DMKNLE	DMKOPR	DMKPAG	DMKPGS	DMKPGT	DMKPRG	DMKPRV	DMKPSA	DMKPTR
		DMKQCN	DMKRGA	DMKRGB	DMKRNH	DMKRPA	DMKRSE	DMKRSP	DMKSAV	DMKSCH	DMKSCN	DMKSEP	DMKSEV
		DMKSIX	DMKSNC	DMKSPL	DMKSSP	DMKSSS	DMKSTK	DMKSVC	DMKTAP	DMKTCS	DMKTDK	DMKTHI	DMKTMR
		DMKTRA	DMKTRC	DMKTRD	DMKTRK	DMKTRM	DMKUDR	DMKUDU	DMKUNT	DMKUSO	DMKVAT	DMKVCA	DMKVCH
		DMKVCN	DMKVDA	DMKVDC	DMKVDD	DMKVDE	DMKVDR	DMKVDS	DMKVER	DMKVIO	DMKVMA	DMKVMC	DMKVMI
		DMKVSI	DMKVSP	DMKVSQ	DMKWRM								
PSACPXBP	000007	DMKCPI	DMKCPU	DMKSVC									
PSAMSS	000009	DMKCFP	DMKCPI	DMKDEF	DMKDGD	DMKLNK	DMKSSS	DMKUSO	DMKVSI				
PSARSV6	000001	DMKPSA											
PSASVCCT	000004	DMKMIA	DMKSVC										
PSBCLR2	000002	DMKCPI											
PSECLR2	000001	DMKCPI											
PSENDCLR	000003	DMKAPI	DMKCPI										
PSTARTSV	000004	DMKSAV											
PSW	000002	DMKLD00E											
PWDCHAIN	000005	DMKJRL											
PWDDATE	000003	DMKJRL											
PWDIBLOK	000003	DMKJRL											
PWDINVCT	000007	DMKJRL											

LABEL	COUNT	REFERENCES											
PWDLOG	000002	DMKJRL											
PWDSIZE	000003	DMKJRL											
PWDTERMA	000002	DMKJRL											
PWDUSRID	000002	DMKJRL											
PWTPAGES	000003	DMKCPU	DMKDSP										
QUANTUM	000009	DMKDSP	DMKEXT	DMKSVC									
QUANTUMR	000010	DMKDSP	DMKEXT	DMKIOS	DMKMCH	DMKPRG	DMKPSA	DMKSVC					
QUEUE	000001	DMKRGB											
Q1DROP	000003	DMKMON	DMKSCH										
RA	000025	DMKDDR	DMKDIR	DMKFMT	DMKGRF	DMKGRT	DMKGRW	DMKRGA	DMKRGB	DMKSSP			
RANGE	000079	DMKCDB	DMKCDM	DMKCPS	DMKCQG	DMKCQH	DMKCQP	DMKMCD	DMKVDA	DMKVDC	DMKVDD		
RCHADD	000019	DMKCCH	DMKCKP	DMKCPI	DMKDSB	DMKENT	DMKIOS	DMKMNI	DMKMON	DMKSCN	DMKSSS		
RCHBLOK	000036	DMKCCH	DMKCFO	DMKCKP	DMKCPB	DMKCPI	DMKCPS	DMKCPV	DMKCQP	DMKDIA	DMKDSB	DMKENT	DMKIOG
		DMKIOS	DMKMNI	DMKMON	DMKNES	DMKPRV	DMKSCN	DMKSSP	DMKSSS	DMKVCH			
RCHBMX	000002	DMKIOS											
RCHBUSY	000015	DMKIOS											
RCHCUTBL	000020	DMKCCH	DMKCKP	DMKCPI	DMKCPS	DMKCPV	DMKCQP	DMKDIA	DMKMNI	DMKNES	DMKSCN	DMKSSP	DMKVCH
RCHDED	000003	DMKCCH	DMKVCH										
RCHFIOB	000007	DMKIOS											
RCHMPX	000003	DMKIOS											
RCHQCNT	000010	DMKENT	DMKIOS	DMKMON									
RCHRSTQ	000006	DMKIOS											
RCHSEL	000001	DMKIOS											
RCHSIZE	000001	DMKSSP											
RCHSTAT	000019	DMKCCH	DMKIOS	DMKVCH									
RCHSTIDC	000003	DMKCCH	DMKIOG	DMKPRV									
RCHTYPE	000009	DMKIOG	DMKIOS										
RCH370	000003	DMKIOG	DMKIOS										
RCUADD	000021	DMKCCH	DMKCKP	DMKCPI	DMKDSB	DMKENT	DMKIOS	DMKMNI	DMKMCN	DMKSCN	DMKSSP	DMKSSS	
RCUBLOK	000064	DMKCCH	DMKCCW	DMKCFO	DMKCKP	DMKCPB	DMKCPI	DMKCPS	DMKCPV	DMKCQP	DMKDIA	DMKDSB	DMKENT
		DMKGRF	DMKIOC	DMKIOS	DMKMNI	DMKMON	DMKNES	DMKNLD	DMKPRV	DMKSCN	DMKSSP	DMKSSS	DMKVCH
RCUBUSY	000016	DMKENT	DMKIOS										
RCUCHA	000034	DMKCFO	DMKCKP	DMKCPB	DMKCPI	DMKCPS	DMKCQP	DMKDSB	DMKENT	DMKIOS	DMKMON	DMKPRV	DMKSCN
		DMKSSP	DMKSSS	DMKVCH									
RCUCHAOF	000009	DMKCFO	DMKCPI	DMKCPS	DMKCQP	DMKIOS	DMKNLD	DMKVCH					
RCUCHB	000002	DMKCCW	DMKSSP										
RCUCHBOF	000007	DMKCFO	DMKCPI	DMKCPS	DMKCQP	DMKIOS	DMKVCH						
RCUCHC	000002	DMKSCN	DMKSSP										
RCUCHCOF	000007	DMKCFO	DMKCPI	DMKCPS	DMKCQP	DMKIOS	DMKVCH						
RCUCHD	000008	DMKCFO	DMKCPI	DMKCPS	DMKCQP	DMKIOS	DMKSSP	DMKVCH					
RCUCHDOF	000006	DMKCFO	DMKCPI	DMKCPS	DMKCQP	DMKIOS	DMKVCH						
RCUDISA	000015	DMKCFO	DMKCPI	DMKCPS	DMKCQP	DMKIOS	DMKNES	DMKNLD	DMKVCH				
RCUDVTBL	000021	DMKCCH	DMKCKP	DMKCPI	DMKCPS	DMKCPV	DMKCQP	DMKDIA	DMKGRF	DMKMNI	DMKNES	DMKNLD	DMKSCN
		DMKSSP	DMKVCH										
RCUFIOB	000006	DMKIOS											
RCUPRIME	000030	DMKCCW	DMKCKP	DMKCPB	DMKCPI	DMKCPS	DMKCQP	DMKDSB	DMKENT	DMKGRF	DMKIOS	DMKMNI	DMKMON

LABEL	COUNT	REFERENCES											
		DMKPRV	DMKSCN	DMKSSS									
RCUQCNT	000014	DMKENT	DMKIOS	DMKMON									
RCURSTQ	000006	DMKIOS											
RCUSCED	000008	DMKIOS											
RCUSHRD	000005	DMKIOS											
RCUSIZE	000002	DMKSSP											
RCUSTAT	000048	DMKCFO	DMKCPI	DMKCPS	DMKCQP	DMKENT	DMKIOS	DMKNES	DMKNLD	DMKVCH			
RCUSUB	000030	DMKCCW	DMKCKP	DMKCPB	DMKCPI	DMKCPS	DMKCQP	DMKDSB	DMKENT	DMKGRF	DMKIOS	DMKMNI	DMKMON
		DMKPRV	DMKSCN	DMKSSS									
RCUTYPE	000040	DMKCCW	DMKCKP	DMKCPB	DMKCPI	DMKCPS	DMKCQP	DMKDSB	DMKENT	DMKGRF	DMKIOC	DMKIOS	DMKMNI
		DMKMON	DMKPRV	DMKSCN	DMKSSP	DMKSSS							
RCU2701	000002	DMKIOC											
RCU2702	000002	DMKIOC											
RCWADDR	000054	DMKCCW	DMKDGD	DMKDIB	DMKHVC	DMKTRK	DMKUNT	DMKVCA					
RCWCCNT	000015	DMKCCW	DMKCFP	DMKCPB	DMKIOS	DMKISM	DMKTRK	DMKUNT	DMKVDR				
RCWCCW	000049	DMKCCW	DMKCFP	DMKCPB	DMKDGD	DMKDIB	DMKHVC	DMKISM	DMKTRD	DMKTRK	DMKUNT	DMKVCA	DMKVDR
RCWCNT	000015	DMKCCW	DMKDGD	DMKDIB	DMKTRK	DMKUNT	DMKVCA						
RCWCOMND	000064	DMKCCW	DMKDGD	DMKDIB	DMKTRK	DMKUNT	DMKVCA						
RCWCTL	000042	DMKCCW	DMKDGD	DMKDIB	DMKHVC	DMKTRK	DMKUNT	DMKVCA					
RCWFLAG	000089	DMKCCW	DMKDGD	DMKDIB	DMKTRK	DMKUNT	DMKVCA						
RCWGEN	000011	DMKCCW	DMKTRD	DMKTRK	DMKUNT								
RCWHEAD	000010	DMKCCW	DMKCFP	DMKCPB	DMKHVC	DMKTRK	DMKVDR						
RCWHMR	000006	DMKCCW	DMKUNT										
RCWINVL	000009	DMKCCW	DMKDIB	DMKTRK	DMKVCA								
RCWIO	000012	DMKCCW	DMKDGD	DMKISM	DMKUNT								
RCWISAM	000001	DMKCCW											
RCWPNT	000017	DMKCCW	DMKHVC	DMKISM	DMKTRD	DMKTRK	DMKUNT						
RCWRCNT	000012	DMKCCW	DMKCFP	DMKHVC	DMKISM	DMKTRD	DMKTRK	DMKUNT					
RCWREL	000012	DMKCCW	DMKTRK										
RCWSHR	000007	DMKCCW	DMKDGD	DMKUNT									
RCWTASK	000048	DMKCCW	DMKCFP	DMKCPB	DMKHVC	DMKIOS	DMKISM	DMKTRD	DMKTRK	DMKUNT	DMKVDR		
RCWVCAW	000010	DMKCCW	DMKISM	DMKTRD	DMKUNT								
RCWVCNT	000004	DMKCCW	DMKTRD										
RCW2311	000005	DMKCCW	DMKUNT										
RDBUFLN	000004	DMKRNH											
RDBUFNO	000003	DMKRNH											
RDCCW	000002	DMKLD00E											
RDCOUNT	000001	DMKDAS											
RDEVACNT	000015	DMKACO	DMKCKP	DMKCQP	DMKCSO	DMKRSE	DMKRSP	DMKSPL					
RDEVACTV	000024	DMKCNS	DMKDIA	DMKGRF	DMKQCN								
RDEVADD	000039	DMKCCH	DMKCKP	DMKCPI	DMKCPS	DMKCQP	DMKDEF	DMKDIA	DMKDSB	DMKENT	DMKIOS	DMKLOG	DMKMNI
		DMKMON	DMKNES	DMKNLD	DMKSCN	DMKSSP	DMKSSS	DMKVCH	DMKVDA	DMKVDC	DMKVDD	DMKVDR	
RDEVAIOB	000040	DMKCCH	DMKCFP	DMKCKP	DMKCNS	DMKCPI	DMKCPS	DMKCSO	DMKDIA	DMKGRF	DMKIOS	DMKLOG	DMKNLD
		DMKRGB	DMKRSE	DMKVSI									
RDEVAIRA	000010	DMKDIA	DMKGRF										
RDEVALLN	000018	DMKCKS	DMKCPI	DMKMON	DMKPGT	DMKTDK	DMKVDC	DMKWRM					

LABEL	COUNT	REFERENCES											
RDEVALT	000007	DMKCPI	DMKIOS	DMKVSI									
RDEVAPLP	000010	DMKCFT	DMKCQR	DMKGRF	DMKQCN								
RDEVATNC	000004	DMKCNS											
RDEVATOF	000008	DMKCFT	DMKCNS	DMKCPI	DMKCQR	DMKTRM							
RDEVATT	000014	DMKCQP	DMKDEF	DMKDIA	DMKIOS	DMKNLD	DMKSSS	DMKVCH	DMKVDD	DMKVDR	DMKVDS		
RDEVAUTO	000008	DMKCKP	DMKCPI	DMKCQP	DMKNET	DMKNLE	DMKRNH	DMKWRM					
RDEVBACK	000010	DMKCSO	DMKRSE	DMKRSP									
RDEVBASE	000009	DMKCCW	DMKCPV	DMKDIA	DMKNES	DMKNET	DMKNLD	DMKPSA					
RDEVBLOK	000286	DMKACO	DMKATS	DMKBLD	DMKBSC	DMKCCH	DMKCCW	DMKCFC	DMKCFG	DMKCFH	DMKCFM	DMKCFO	DMKCFP
		DMKCFT	DMKCKP	DMKCKS	DMKCNS	DMKCPB	DMKCPI	DMKCPS	DMKCPU	DMKCPV	DMKCQG	DMKCQP	DMKCQR
		DMKCQY	DMKCSB	DMKCSO	DMKDAS	DMKDEF	DMKDGD	DMKDIA	DMKDMP	DMKDRD	DMKDSB	DMKDSP	DMKENT
		DMKGRF	DMKGRT	DMKHVD	DMKIOC	DMKIOE	DMKIOF	DMKIOG	DMKIOS	DMKLNK	DMKLOG	DMKLOH	DMKMCC
		DMKMNI	DMKMON	DMKMSW	DMKNES	DMKNET	DMKNLD	DMKNLE	DMKOPR	DMKPAG	DMKPGS	DMKPGT	DMKPRV
		DMKPSA	DMKPTR	DMKQCN	DMKRGA	DMKRGB	DMKRNH	DMKRSE	DMKRSP	DMKSCN	DMKSEP	DMKSNC	DMKSPL
		DMKSSP	DMKSSS	DMKTAP	DMKTCS	DMKTDK	DMKTRK	DMKTRM	DMKUNT	DMKUSO	DMKVCH	DMKVCN	DMKVDA
		DMKVDC	DMKVDD	DMKVDE	DMKVDR	DMKVDS	DMKVER	DMKVSI	DMKWRM				
RDEVBSC	000009	DMKBSC	DMKRGA	DMKRGB									
RDEVBUCH	000009	DMKIOS											
RDEVBUSY	000025	DMKACO	DMKCCH	DMKCPB	DMKCPS	DMKCSO	DMKENT	DMKIOS	DMKRNH	DMKRSE	DMKRSP	DMKVCH	DMKVDA
RDEVCKPT	000007	DMKRNH	DMKWRM										
RDEVCLAS	000016	DMKACO	DMKCFP	DMKCKP	DMKCKS	DMKCQP	DMKCSC	DMKRSP	DMKSPL	DMKSSP	DMKWRM		
RDEVCODE	000020	DMKATS	DMKCFG	DMKCFH	DMKCKS	DMKCPI	DMKCPU	DMKHVD	DMKIOG	DMKNLD	DMKPGT	DMKSNC	DMKTCS
		DMKVDC	DMKWRM										
RDEVCON	000058	DMKCNS	DMKDIA	DMKGRF	DMKNES	DMKQCN	DMKRGA	DMKRGB	DMKRNH				
RDEVCONC	000003	DMKIOS											
RDEVCORD	000015	DMKDIA	DMKGRF	DMKOPR									
RDEVCORR	000002	DMKCNS	DMKTRM										
RDEVCPNA	000006	DMKGRF											
RDEVCTL	000029	DMKCNS	DMKDIA	DMKGRF									
RDEVCTRS	000016	DMKCPS	DMKIOE	DMKIOF	DMKNES	DMKNET							
RDEVCUA	000030	DMKCCW	DMKCKP	DMKCPB	DMKCPI	DMKCPS	DMKDIA	DMKDSB	DMKENT	DMKGRF	DMKIOC	DMKIOS	DMKMNI
		DMKMON	DMKNES	DMKNLD	DMKPRV	DMKSCN	DMKSSP	DMKSSS					
RDEVCUB	000011	DMKCCW	DMKCPI	DMKDAS	DMKDSB	DMKIOS	DMKSCN	DMKTAP					
RDEVCURP	000003	DMKTCS	DMKVDR										
RDEVCYL	000006	DMKDIA	DMKIOS	DMKMON	DMKPGT								
RDEVDED	000071	DMKACO	DMKCFO	DMKCKP	DMKCPS	DMKCPV	DMKCQP	DMKCSB	DMKCSO	DMKDAS	DMKDEF	DMKDIA	DMKDSB
		DMKGRF	DMKIOS	DMKLOG	DMKMCC	DMKMSW	DMKNES	DMKNET	DMKNLD	DMKNLE	DMKRGB	DMKRNH	DMKRSP
		DMKSCN	DMKSPL	DMKSSS	DMKVCH	DMKVDA	DMKVDC	DMKVDD	DMKVDR	DMKVDS	DMKVER		
RDEVDELP	000014	DMKCKP	DMKCPS	DMKRSE	DMKRSP	DMKVDR	DMKVDS						
RDEVDISA	000078	DMKACO	DMKCFO	DMKCKP	DMKCKS	DMKCNS	DMKCPI	DMKCPS	DMKCPV	DMKCQP	DMKCSB	DMKCSO	DMKDAS
		DMKDEF	DMKDSB	DMKGRF	DMKIOS	DMKLOG	DMKMCC	DMKMNI	DMKNES	DMKNET	DMKNLD	DMKNLE	DMKRGA
		DMKRGB	DMKRNH	DMKRSP	DMKSCN	DMKSPL	DMKSSS	DMKVCH	DMKVDA	DMKVDC	DMKVDD	DMKVDE	DMKVDS
		DMKWRM											
RDEVDISB	000026	DMKCKP	DMKCNS	DMKCPV	DMKGRF	DMKNES	DMKNET	DMKRGA	DMKRGB				
RDEVDRAN	000036	DMKACO	DMKCKP	DMKCKS	DMKCPS	DMKCQP	DMKCSB	DMKCSO	DMKRSP	DMKSPL	DMKVCH	DMKVDA	DMKVDC
		DMKVDS	DMKWRM										

LABEL	COUNT	REFERENCES											
RDEVDROP	000007	DMKCNS	DMKGRF	DMKQCN									
RDEVENAB	000039	DMKCKP	DMKCNS	DMKCPI	DMKCPS	DMKCPV	DMKCQP	DMKGRF	DMKNES	DMKNET	DMKNLD	DMKRGA	DMKVCH
		DMKVDA	DMKVDC	DMKVDS	DMKWRM								
RDEVEPDV	000018	DMKDIA	DMKNES	DMKNLD									
RDEVEPLN	000005	DMKCPS	DMKDIA	DMKNES	DMKNLD	DMKVDS							
RDEVEPMD	000007	DMKCNS	DMKDIA	DMKNES	DMKNLD								
RDEVEXTN	000008	DMKCPI	DMKTCS										
RDEVFIOB	000014	DMKCPS	DMKDSP	DMKIOS	DMKNLD	DMKPGT	DMKSSS						
RDEVFLAG	000351	DMKACO	DMKBLD	DMKCFG	DMKCFH	DMKCFO	DMKCFT	DMKCKP	DMKCKS	DMKCNS	DMKCPI	DMKCPS	DMKCPV
		DMKCQP	DMKCQR	DMKCSB	DMKCSO	DMKDAS	DMKDEF	DMKDIA	DMKDSB	DMKGRF	DMKHVD	DMKIOS	DMKLOG
		DMKMCC	DMKMNI	DMKMON	DMKNES	DMKNET	DMKNLD	DMKNLE	DMKPGT	DMKPSA	DMKQCN	DMKRGA	DMKRGB
		DMKRNH	DMKRSE	DMKRSP	DMKSCN	DMKSEP	DMKSNC	DMKSPL	DMKSSS	DMKTCS	DMKTRM	DMKVCH	DMKVDA
		DMKVDC	DMKVDD	DMKVDR	DMKVDS	DMKWRM							
RDEVFSEP	000011	DMKCKP	DMKCKS	DMKCQP	DMKCSO	DMKSSP	DMKTCS	DMKWRM					
RDEVFTR	000120	DMKACO	DMKBSC	DMKCCW	DMKCFP	DMKCKP	DMKCPI	DMKCQP	DMKCSB	DMKDAS	DMKDEF	DMKDGD	DMKDSB
		DMKHVD	DMKIOE	DMKIOF	DMKIOS	DMKLNK	DMKLOG	DMKNLD	DMKNLE	DMKPAG	DMKPGT	DMKRGA	DMKRGB
		DMKRSE	DMKSPL	DMKSSP	DMKSSS	DMKTAP	DMKTCS	DMKTDK	DMKUNT	DMKVDA	DMKVDC	DMKVDE	DMKVDS
		DMKVER	DMKVSI										
RDEVGRTB	000005	DMKGRF											
RDEVGRTY	000009	DMKGRF	DMKGRT	DMKOPR	DMKQCN	DMKVCN							
RDEVHIO	000017	DMKCNS	DMKDIA	DMKGRF	DMKPSA								
RDEVHOLD	000008	DMKGRF											
RDEVIDNT	000008	DMKCNS	DMKCPI	DMKTRM									
RDEVIMAG	000012	DMKCKP	DMKCKS	DMKCQP	DMKCSO	DMKSSP	DMKTCS	DMKWRM					
RDEVIOBL	000005	DMKIOS	DMKPAG										
RDEVIOCT	000005	DMKIOS	DMKMNI	DMKMON									
RDEVIOER	000040	DMKBSC	DMKCFP	DMKCNS	DMKCPS	DMKCSO	DMKDAS	DMKGRF	DMKIOE	DMKIOS	DMKMSW	DMKRSE	DMKRSP
		DMKTAP	DMKTRK										
RDEVIRM	000007	DMKCFO	DMKIOE	DMKNES	DMKNLD								
RDEVLCEP	000010	DMKCKP	DMKCPS	DMKCQP	DMKDIA	DMKNLD	DMKRNH						
RDEVLIOB	000002	DMKIOS											
RDEVLLZN	000006	DMKBLD	DMKCFT	DMKCQR	DMKHVD	DMKQCN							
RDEVLNCP	000021	DMKCKP	DMKCPS	DMKCQP	DMKDIA	DMKNES	DMKNET	DMKNLD	DMKRNH	DMKVDS			
RDEVLNKS	000017	DMKCPS	DMKCQP	DMKDEF	DMKSCN	DMKSSS	DMKTDK	DMKVDA	DMKVDC	DMKVDR	DMKVDS		
RDEVLOAD	000005	DMKCSO	DMKRSP	DMKSEP									
RDEVLOG	000010	DMKCNS	DMKCPV	DMKGRF									
RDEVMAX	000028	DMKCKP	DMKCPI	DMKCPS	DMKNES	DMKNET	DMKNLD	DMKRGA	DMKRGB	DMKRNH	DMKVDS	DMKWRM	
RDEVMAXP	000002	DMKRSP	DMKSSP										
RDEVMDL	000025	DMKACO	DMKCKP	DMKCPI	DMKHVD	DMKIOC	DMKIOF	DMKNES	DMKNLD	DMKNLE	DMKPAG	DMKUNT	DMKVDC
		DMKVSI											
RDEVMORE	000008	DMKGRF											
RDEVMOUT	000016	DMKCPS	DMKCQP	DMKDAS	DMKDSB	DMKSCN	DMKSSS	DMKVDA	DMKVDC	DMKVDR	DMKVDS		
RDEVNAME	000001	DMKIOC											
RDEVNCP	000010	DMKCKP	DMKCQP	DMKNLD	DMKRNH	DMKWRM							
RDEVNICL	000056	DMKACO	DMKCFT	DMKCKP	DMKCPI	DMKCPS	DMKCQR	DMKDIA	DMKHVD	DMKLOG	DMKNES	DMKNET	DMKNLD
		DMKPSA	DMKQCN	DMKRGA	DMKRGB	DMKRNH	DMKVDS	DMKWRM					

LABEL	COUNT	REFERENCES											
RDEVNRDY	000048	DMKCNS	DMKCPI	DMKCPS	DMKCQP	DMKCSO	DMKDAS	DMKDIA	DMKIOS	DMKNES	DMKNET	DMKNLD	DMKNLE
		DMKRGA	DMKRGB	DMKRNH	DMKRSE	DMKRSP	DMKTAP	DMKVDE					
RDEVOVLY	000004	DMKCQP	DMKCSO	DMKRSP	DMKVDR								
RDEVOWN	000026	DMKCFG	DMKCFH	DMKCPI	DMKCPS	DMKCQP	DMKDAS	DMKDSB	DMKHVC	DMKLOG	DMKNLD	DMKSNC	DMKSSS
		DMKTCS	DMKVCH	DMKVDA	DMKVDC	DMKVDD	DMKVDS						
RDEVPAGE	000008	DMKPGT	DMKUSO										
RDEVPDLY	000003	DMKNES	DMKRGA										
RDEVPNT	000014	DMKCPI	DMKPGT	DMKTDK	DMKUSO	DMKVDC							
RDEVPREF	000011	DMKCKS	DMKCPI	DMKPGT	DMKVDA	DMKVDC							
RDEVPREP	000015	DMKCNS	DMKDIA	DMKVDA									
RDEVPSUP	000015	DMKBLD	DMKCFT	DMKCNS	DMKCQR	DMKLOG							
RDEVPTTC	000006	DMKCNS	DMKCPI	DMKNES	DMKNLD	DMKTRM							
RDEVPURG	000005	DMKCQP	DMKCSO	DMKRSP	DMKTCS								
RDEVQCNT	000010	DMKENT	DMKIOS	DMKMON									
RDEVRACT	000012	DMKIOS											
RDEVRCNT	000013	DMKCNS											
RDEVRCVY	000016	DMKCPS	DMKDIA	DMKNES	DMKNLD	DMKNLE	DMKRNH	DMKVCH	DMKVDC	DMKVDS			
RDEVREAD	000008	DMKGRF											
RDEVRECS	000010	DMKCKP	DMKCKS	DMKDMP	DMKPGT	DMKUSO	DMKWRM						
RDEVREST	000004	DMKCNS											
RDEVRSTR	000011	DMKCSO	DMKRSE	DMKRSP									
RDEVRSVD	000018	DMKCPS	DMKNES	DMKNET	DMKNLD	DMKNLE	DMKRGA	DMKRGB	DMKRNH	DMKVCH	DMKVDC	DMKVDS	
RDEVRUN	000011	DMKCPI	DMKDIA	DMKGRF									
RDEVSADN	000005	DMKCCW	DMKCNS	DMKIOC	DMKNES								
RDEVSCED	000014	DMKCPS	DMKIOS	DMKRNH	DMKVCH								
RDEVSEL	000007	DMKDSB	DMKSSS										
RDEVSEP	000012	DMKCKP	DMKCQP	DMKCSO	DMKRSP	DMKSEP	DMKWRM						
RDEVSER	000055	DMKCFG	DMKCKS	DMKCPI	DMKCPS	DMKCQG	DMKCQE	DMKDAS	DMKDEF	DMKDSB	DMKIOE	DMKLOG	DMKMNI
		DMKMON	DMKSCN	DMKSSS	DMKVDA	DMKVDD	DMKVDE	DMKVER	DMKWRM				
RDEVSIZE	000005	DMKLOG	DMKSCN	DMKSSP	DMKSSS								
RDEVSKUP	000004	DMKIOS	DMKMON										
RDEVSLOW	000009	DMKCQP	DMKNES	DMKRNH									
RDEVSPAC	000004	DMKCSO	DMKRSP										
RDEVSPL	000038	DMKACO	DMKCKP	DMKCKS	DMKCPS	DMKCQP	DMKCSB	DMKCSO	DMKRSE	DMKRSP	DMKSPL	DMKVCH	DMKVDA
		DMKVDC	DMKVDS	DMKWRM									
RDEVSTAT	000237	DMKACO	DMKCCH	DMKCFO	DMKCKP	DMKCKS	DMKCNS	DMKCPB	DMKCPI	DMKCPS	DMKCPV	DMKCQP	DMKCSB
		DMKCSO	DMKDAS	DMKDEF	DMKDIA	DMKDSB	DMKENT	DMKGRF	DMKIOE	DMKIOS	DMKLOG	DMKMCC	DMKMNI
		DMKMSW	DMKNES	DMKNET	DMKNLD	DMKNLE	DMKRGA	DMKRGB	DMKRNH	DMKRSE	DMKRSP	DMKSCN	DMKSPL
		DMKSSS	DMKTAP	DMKVCH	DMKVDA	DMKVDC	DMKVDD	DMKVDE	DMKVDR	DMKVDS	DMKVER	DMKWRM	
RDEVSTA2	000056	DMKCNS	DMKCPI	DMKCQP	DMKCSO	DMKGRF	DMKIOS	DMKNLD	DMKQCN	DMKRSP	DMKTCS	DMKVSI	
RDEVSYNC	000007	DMKCNS											
RDEVSYS	000027	DMKCFO	DMKCPI	DMKCPS	DMKCQP	DMKDAS	DMKDSB	DMKLOG	DMKMCC	DMKMNI	DMKSSS	DMKVCH	DMKVDA
		DMKVDC	DMKVDD	DMKVDR	DMKVDS								
RDEVTBTU	000003	DMKNES	DMKRNH										
RDEVTCTL	000002	DMKNES											
RDEVTERM	000015	DMKCSO	DMKRSE	DMKRSP									

LABEL	COUNT	REFERENCES											
RDEVTEXT	000012	DMKCFT	DMKCQR	DMKGRF									
RDEVTFLG	000084	DMKCFT	DMKCNS	DMKCPI	DMKCPV	DMKCQR	DMKDIA	DMKGRF	DMKNLD	DMKTRM			
RDEVTMAT	000006	DMKACO	DMKCKP	DMKDIA	DMKVDR	DMKVDS							
RDEVTMCD	000034	DMKCFT	DMKCNS	DMKCPI	DMKCQR	DMKGRF	DMKHVD	DMKIOC	DMKNES	DMKNLD	DMKQCN	DMKTRM	
RDEVTRQ	000005	DMKGRF											
RDEVTYPC	000337	DMKACO	DMKBLD	DMKCCW	DMKCFM	DMKCFO	DMKCFT	DMKCKP	DMKCNS	DMKCPI	DMKCPS	DMKCPV	DMKCQG
		DMKCQP	DMKCQR	DMKCQY	DMKCSB	DMKCSO	DMKDEF	DMKDIA	DMKDMP	DMKGRF	DMKHVD	DMKIOC	DMKIOE
		DMKIOF	DMKIOS	DMKLNK	DMKLOG	DMKLOH	DMKMCC	DMKMNI	DMKMCN	DMKMSW	DMKNES	DMKNET	DMKNLD
		DMKNLE	DMKOPR	DMKPSA	DMKQCN	DMKRGA	DMKRNH	DMKRSE	DMKRSP	DMKSCN	DMKSSP	DMKSSS	DMKUSO
		DMKVCH	DMKVCN	DMKVDA	DMKVDC	DMKVDD	DMKVDE	DMKVDR	DMKVDS	DMKVER	DMKVSI	DMKWRM	
RDEVTYPE	000469	DMKACO	DMKATS	DMKBLD	DMKCCW	DMKCFC	DMKCFG	DMKCFH	DMKCFM	DMKCFO	DMKCFP	DMKCFT	DMKCKP
		DMKCKS	DMKCNS	DMKCPI	DMKCPS	DMKCPU	DMKCPV	DMKCQG	DMKCQP	DMKCQR	DMKCQY	DMKCSB	DMKCSO
		DMKDAS	DMKDIA	DMKDMP	DMKDRD	DMKDSB	DMKGRF	DMKHVD	DMKIOC	DMKIOE	DMKIOF	DMKIOG	DMKIOS
		DMKLNK	DMKLOG	DMKLOH	DMKMON	DMKMSW	DMKNES	DMKNLD	DMKNLE	DMKOPR	DMKPAG	DMKPGS	DMKPGT
		DMKPTR	DMKQCN	DMKRGA	DMKRSE	DMKRSP	DMKSEP	DMKSNC	DMKSPL	DMKSSP	DMKSSS	DMKTAP	DMKTCS
		DMKTDK	DMKUSO	DMKVCH	DMKVCN	DMKVDA	DMKVDC	DMKVDD	DMKVDE	DMKVDR	DMKVDS	DMKVER	DMKWRM
RDEVUSC8	000002	DMKCNS	DMKNES										
RDEVUSER	000041	DMKBLD	DMKCCW	DMKCFO	DMKCNS	DMKCPI	DMKCPS	DMKCPV	DMKCQP	DMKCSB	DMKCSO	DMKDIA	DMKGRF
		DMKIOS	DMKLOG	DMKMCC	DMKNES	DMKNET	DMKNLD	DMKNLE	DMKPSA	DMKSSS	DMKVDA	DMKVDD	DMKVDR
		DMKVDS											
RDEVWAII	000014	DMKRGA	DMKRGB										
RDEVWAIT	000008	DMKNES	DMKRNH										
RDEVXSEP	000009	DMKCKP	DMKCKS	DMKCQP	DMKCSO	DMKSSP	DMKTCS	DMKWRM					
RDEV333V	000005	DMKIOS	DMKLOG	DMKVDA									
RDRCHN	000013	DMKCKS	DMKDMP	DMKMIA	DMKNLE	DMKSPL	DMKWRM						
READ	000024	DMKDIR	DMKLD00E	DMKLNK	DMKSPL	DMKVSP							
READBUF	000005	DMKDSB	DMKGRF	DMKRNH	DMKRSE								
READER1	000002	DMKVSP											
READNRM	000002	DMKRNH											
RECBLOK	000053	DMKCKP	DMKCKS	DMKCPI	DMKDMP	DMKPGT	DMKRSP	DMKSPL	DMKVSP	DMKWRM			
RECCCPD	000004	DMKIOF	DMKIOG										
RECCYL	000031	DMKCKP	DMKCKS	DMKCPI	DMKDMP	DMKPGT	DMKRSP	DMKVSP	DMKWRM				
RECFLAG1	000018	DMKIOF	DMKIOG										
RECFLAG2	000004	DMKIOG											
RECMAP	000037	DMKCKS	DMKCPI	DMKDMP	DMKPGT	DMKRSP	DMKVSP						
RECMAX	000012	DMKCKP	DMKCKS	DMKCPI	DMKPGT								
RECMODE	000005	DMKIOG	DMKMCH										
RECNXT	000013	DMKIOF	DMKIOG										
RECOVRPT	000006	DMKMCH											
RECPAG	000006	DMKIOF	DMKIOG										
RECPAGDN	000001	DMKIOG											
RECPAGFA	000007	DMKIOG											
RECPAGFL	000006	DMKIOF	DMKIOG										
RECPAGFM	000003	DMKIOG											
RECPAGFR	000002	DMKIOG											
RECPAGIU	000003	DMKIOF	DMKIOG										

LABEL	COUNT	REFERENCES											
RECPNT	000032	DMKCKP	DMKCKS	DMKCPI	DMKDMP	DMKPGT	DMKRSP	DMKSPL	DMKVSP	DMKWRM			
RECSIZE	000019	DMKCKP	DMKCKS	DMKCPI	DMKPGT	DMKRSP	DMKSPL	DMKUSO	DMKVSP	DMKWRM			
RECUSED	000025	DMKCKP	DMKCKS	DMKCPI	DMKDMP	DMKPGT	DMKRSP	DMKVSP	DMKWRM				
REGSAV	000004	DMKFMT											
RESET	000004	DMKCFC	DMKEXT	DMKFMT	DMKMCT								
RESTDEV	000001	DMKCCH											
RETREG	000002	DMKLD00E											
RETRYSW	000003	DMKCNS											
RETT	000003	DMKLD00E											
RSPBF1IO	000005	DMKRSP											
RSPBF1VL	000006	DMKRSP											
RSPBF2IO	000004	DMKRSP											
RSPBF2VL	000005	DMKRSP											
RSPDPAGE	000019	DMKRSP	DMKSPL	DMKTCS									
RSPDPAG2	000007	DMKRSP											
RSPFLAG1	000018	DMKRSP											
RSPLCTL	000014	DMKCKP	DMKCQP	DMKCSO	DMKRSP	DMKSPL	DMKTCS						
RSPMISC	000006	DMKCSO	DMKRSP										
RSPRPAGE	000032	DMKRSP	DMKSPL	DMKTCS									
RSPRPAG2	000007	DMKRSP	DMKTCS										
RSPRSTRT	000007	DMKRSP											
RSPSPBLK	000018	DMKCKP	DMKCQP	DMKCSO	DMKRSP	DMKSPL							
RSPSIZE	000005	DMKRSP	DMKSPL										
RSPVPAGE	000013	DMKRSP	DMKSPL	DMKTCS									
RSPVPAG2	000011	DMKRSP											
RSPXBLOK	000001	DMKTCS											
RSPXCHR	000002	DMKTCS											
RSPXCMOD	000002	DMKTCS											
RSPXCRWC	000001	DMKTCS											
RSPXFCB	000002	DMKTCS											
RSPXSIZE	000002	DMKCPI											
RSRTNPSW	000005	DMKAPI	DMKDMP	DMKMCT									
RTCODE0	000002	DMKEIG											
RTCODE1	000004	DMKEIG	DMKSEV	DMKSIX									
RTCODE2	000004	DMKEIG	DMKSEV	DMKSIX									
RTCODE3	000007	DMKEIG	DMKSEV	DMKSIX									
RTCODE4	000010	DMKEIG	DMKSEV	DMKSIX									
RTCODE5	000004	DMKEIG	DMKSEV	DMKSIX									
RTCODE7	000004	DMKSEV											
RUN	000002	DMKCFC	DMKTRA										
RUNCR0	000012	DMKAPI	DMKCPI	DMKDSP	DMKEXT	DMKPRG	DMKPRV	DMKSVC					
RUNCR1	000003	DMKAPI	DMKCPI	DMKDSP									
RUNPSW	000016	DMKDSP	DMKSVC										
RUNUSER	000036	DMKAPI	DMKCDS	DMKCFP	DMKCPI	DMKDIA	DMKDSP	DMKIOS	DMKLOG	DMKMCH	DMKMCT	DMKPRG	DMKPSA
		DMKSCH	DMKSVC	DMKTHI	DMKUSO	DMKVCA							
R0	007843	DMKACO	DMKALG	DMKAPI	DMKATS	DMKBLD	DMKBSC	DMKCCH	DMKCCW	DMKCDE	DMKCDM	DMKCDS	DMKCFC

LABEL	COUNT	REFERENCES											
		DMKCFD	DMKCFG	DMKCFH	DMKCFM	DMKCFO	DMKCFP	DMKCFS	DMKCFT	DMKCKP	DMKCKS	DMKCLK	DMKCNS
		DMKCPB	DMKCPI	DMKCPS	DMKCPU	DMKCPV	DMKCQG	DMKCQH	DMKCQP	DMKCQR	DMKCQY	DMKCSB	DMKCSO
		DMKCSP	DMKCSQ	DMKCST	DMKCSU	DMKCSV	DMKCVT	DMKDAS	DMKDDR	DMKDEF	DMKDGD	DMKDIA	DMKDIB
		DMKDIR	DMKDMP	DMKDRD	DMKDSB	LMKDSP	DMKEIG	DMKENT	DMKERM	DMKEXT	DMKFMT	DMKFRE	DMKGIO
		DMKGRF	DMKGRT	DMKHVC	DMKHVD	DMKIMG	DMKIOC	DMKIOE	DMKICF	DMKIOG	DMKIOS	DMKISM	DMKJRL
		DMKLD00E	DMKLNK	DMKLOC	DMKLOG	DMKLOH	DMKLOK	DMKMCC	DMKMCD	DMKMCH	DMKMCT	DMKMIA	DMKMID
		DMKMNI	DMKMON	DMKMSG	DMKMSW	DMKNEM	DMKNES	DMKNET	DMKNLD	DMKNLE	DMKNMT	DMKOPR	DMKPAG
		DMKPGS	DMKPGT	DMKPRG	DMKPRV	DMKPSA	DMKPTR	DMKQCN	DMKRGA	DMKRGB	DMKRND	DMKRNH	DMKRPA
		DMKRSE	DMKRSP	DMKSAV	DMKSCH	DMKSCN	DMKSEP	DMKSEV	DMKSIX	DMKSNC	DMKSPL	DMKSSP	DMKSSS
		DMKSTK	DMKSVC	DMKTAP	DMKTCS	DMKTDK	DMKTHI	DMKTMR	DMKTRA	DMKTRC	DMKTRD	DMKTRK	DMKTRM
		DMKUDR	DMKUDU	DMKUNT	DMKUSO	DMKVAT	DMKVCA	DMKVCH	DMKVCN	DMKVDA	DMKVDC	DMKVDD	DMKVDE
		DMKVDR	DMKVDS	DMKVER	DMKVIO	DMKVMA	DMKVMC	DMKVMI	DMKVSI	DMKVSP	DMKVSQ	DMKWRM	
R1	014260	DMKACO	DMKALG	DMKAPI	DMKATS	DMKBLD	DMKBSC	DMKCCH	DMKCCW	DMKCDE	DMKCDM	DMKCDS	DMKCFC
		DMKCFD	DMKCFG	DMKCFH	DMKCFM	DMKCFO	DMKCFP	DMKCFS	DMKCFT	DMKCKP	DMKCKS	DMKCLK	DMKCNS
		DMKCPB	DMKCPI	DMKCPS	DMKCPU	DMKCPV	DMKCQG	DMKCQH	DMKCQP	DMKCQR	DMKCQY	DMKCSB	DMKCSO
		DMKCSP	DMKCSQ	DMKCST	DMKCSU	DMKCSV	DMKCVT	DMKDAS	DMKDDR	DMKDEF	DMKDGD	DMKDIA	DMKDIB
		DMKDIR	DMKDMP	DMKDRD	DMKDSB	DMKDSP	DMKEIG	DMKENT	DMKERM	DMKEXT	DMKFMT	DMKFRE	DMKGIO
		DMKGRF	DMKGRT	DMKHVC	DMKHVD	DMKIMG	DMKIOE	DMKIOF	DMKIOG	DMKIOS	DMKISM	DMKJRL	DMKLD00E
		DMKLNK	DMKLOC	DMKLOG	DMKLOH	DMKLOK	DMKMCC	DMKMCD	DMKMCH	DMKMCT	DMKMIA	DMKMID	DMKMNI
		DMKMON	DMKMSG	DMKMSW	DMKNEM	DMKNES	DMKNET	DMKNLD	DMKNLE	DMKNMT	DMKOPR	DMKPAG	DMKPGS
		DMKPGT	DMKPRG	DMKPRV	DMKPSA	DMKPTR	DMKQCN	DMKRGA	DMKRGB	DMKRND	DMKRNH	DMKRPA	DMKRSE
		DMKRSP	DMKSAV	DMKSCH	DMKSCN	DMKSEP	DMKSEV	DMKSIX	DMKSNC	DMKSPL	DMKSSP	DMKSSS	DMKSTK
		DMKSVC	DMKTAP	DMKTCS	DMKTDK	DMKTHI	DMKTMR	DMKTRA	DMKTRC	DMKTRD	DMKTRK	DMKTRM	DMKUDR
		DMKUDU	DMKUNT	DMKUSO	DMKVAT	DMKVCA	DMKVCH	DMKVCN	DMKVDA	DMKVDC	DMKVDD	DMKVDE	DMKVDR
		DMKVDS	DMKVER	DMKVIO	DMKVMA	DMKVMC	DMKVMI	DMKVSI	DMKVSP	DMKVSQ	DMKWRM		
R10	002529	DMKACO	DMKALG	DMKAPI	DMKATS	DMKBLD	DMKBSC	DMKCCH	DMKCCW	DMKCDE	DMKCDM	DMKCDS	DMKCFD
		DMKCFG	DMKCFH	DMKCFO	DMKCFP	DMKCFS	DMKCFT	DMKCKP	DMKCKS	DMKCNS	DMKCPB	DMKCPI	DMKCPS
		DMKCPU	DMKCPV	DMKCQG	DMKCQH	DMKCQP	DMKCQR	DMKCQY	DMKCSB	DMKCSO	DMKCSP	DMKCSQ	DMKCST
		DMKCSU	DMKCSV	DMKCVT	DMKDAS	DMKDDR	DMKDEF	DMKDGD	DMKDIA	DMKDIB	DMKDIR	DMKDMP	DMKDRD
		DMKDSB	DMKDSP	DMKERM	DMKFMT	DMKFRE	DMKGIC	DMKGRF	DMKHVC	DMKHVD	DMKIOE	DMKIOF	DMKIOG
		DMKIOS	DMKISM	DMKJRL	DMKLD00E	DMKLNK	DMKLOC	DMKLOG	DMKLOH	DMKLOK	DMKMCC	DMKMCD	DMKMCH
		DMKMCT	DMKMIA	DMKMID	DMKMNI	DMKMON	DMKMSG	DMKMSW	DMKNES	DMKNET	DMKNLD	DMKNLE	DMKOPR
		DMKPAG	DMKPGS	DMKPGT	DMKPRG	DMKPRV	DMKPSA	DMKPTR	DMKQCN	DMKRGA	DMKRGB	DMKRND	DMKRNH
		DMKRSE	DMKRSP	DMKSAV	DMKSCH	DMKSCN	DMKSEP	DMKSNC	DMKSPL	DMKSSP	DMKSSS	DMKSTK	DMKSVC
		DMKTAP	DMKTCS	DMKTDK	DMKTHI	DMKTMR	DMKTRC	DMKTRD	DMKTRK	DMKUDR	DMKUDU	DMKUNT	DMKUSO
		DMKVAT	DMKVCA	DMKVCH	DMKVCN	DMKVDA	DMKVDC	DMKVDD	DMKVDE	DMKVDR	DMKVDS	DMKVER	DMKVIO
		DMKVMA	DMKVMC	DMKVMI	DMKVSI	DMKVSP	DMKVSQ	DMKWRM					
R11	001780	DMKACO	DMKALG	DMKAPI	DMKATS	DMKBLD	DMKBSC	DMKCCH	DMKCCW	DMKCDE	DMKCDM	DMKCDS	DMKCFC
		DMKCFD	DMKCFG	DMKCFH	DMKCFM	DMKCFO	DMKCFP	DMKCFS	DMKCFT	DMKCKP	DMKCKS	DMKCLK	DMKCNS
		DMKCPB	DMKCPI	DMKCPS	DMKCPU	DMKCPV	DMKCQG	DMKCQH	DMKCQP	DMKCQR	DMKCQY	DMKCSB	DMKCSO
		DMKCSP	DMKCSQ	DMKCST	DMKCSU	DMKCSV	DMKDAS	DMKDDR	DMKDEF	DMKDGD	DMKDIA	DMKDIB	DMKDIR
		DMKDMP	DMKDRD	DMKDSB	DMKDSP	DMKENT	DMKERM	DMKEXT	DMKFMT	DMKFRE	DMKGIO	DMKGRF	DMKGRT
		DMKHVC	DMKHVD	DMKIOC	DMKIOE	DMKIOF	DMKIOG	DMKIOS	DMKISM	DMKJRL	DMKLNK	DMKLOG	DMKLOH
		DMKLOK	DMKMCC	DMKMCD	DMKMCH	DMKMCT	DMKMIA	DMKMID	DMKMNI	DMKMON	DMKMSG	DMKMSW	DMKNEM
		DMKNES	DMKNET	DMKNLD	DMKNLE	DMKPAG	DMKPGS	DMKPGT	DMKPRG	DMKPRV	DMKPSA	DMKPTR	DMKQCN
		DMKRGA	DMKRGB	DMKRND	DMKRNH	DMKRPA	DMKRSE	DMKRSP	DMKSAV	DMKSCH	DMKSCN	DMKSEP	DMKSNC

LABEL	COUNT	REFERENCES											
		DMKSPL	DMKSSP	DMKSSS	DMKSTK	DMKSVC	DMKTAP	DMKTCS	DMKTDK	DMKTHI	DMKTMR	DMKTRA	DMKTRC
		DMKTRD	DMKTRK	DMKTRM	DMKUDR	DMKUDU	DMKUNT	DMKUSO	DMKVAT	DMKVCA	DMKVCH	DMKVCN	DMKVDA
		DMKVDC	DMKVDD	DMKVDE	DMKVDR	DMKVDS	DMKVER	DMKVIO	DMKVMA	DMKVMC	DMKVMI	DMKVSI	DMKVSP
		DMKVSQ	DMKWRM										
R12	002131	DMKACO	DMKALG	DMKAPI	DMKATS	DMKBLD	DMKBSC	DMKCCH	DMKCCW	DMKCDB	DMKCDM	DMKCDS	DMKCFC
		DMKCFD	DMKCFG	DMKCFH	DMKCFM	DMKCFO	DMKCFP	DMKCFS	DMKCFT	DMKCKP	DMKCKS	DMKCLK	DMKCNS
		DMKCPB	DMKCPI	DMKCPS	DMKCPU	DMKCPV	DMKCQG	DMKCQH	DMKCQP	DMKCQR	DMKCQY	DMKCSB	DMKCSO
		DMKCSP	DMKCSQ	DMKCST	DMKCSU	DMKCSV	DMKDAS	DMKDDR	DMKDEF	DMKDGD	DMKDIA	DMKDIB	DMKDIR
		DMKDMP	DMKDRD	DMKDSB	DMKDSP	DMKEIG	DMKENT	DMKERM	DMKEXT	DMKFMT	DMKFRE	DMKGIO	DMKGRF
		DMKGRT	DMKHVC	DMKHVD	DMKIMG	DMKIOC	DMKIOE	DMKIOF	DMKIOG	DMKIOS	DMKISM	DMKJRL	DMKLD00E
		DMKLNK	DMKLOC	DMKLOG	DMKLOH	DMKLOK	DMKMCC	DMKMCD	DMKMCH	DMKMCT	DMKMIA	DMKMID	DMKMNI
		DMKMON	DMKMSG	DMKMSW	DMKNEM	DMKNES	DMKNET	DMKNLD	DMKNLE	DMKNMT	DMKPAG	DMKPGS	DMKPGT
		DMKPRG	DMKPRV	DMKPSA	DMKPTR	DMKQCN	DMKRGA	DMKRGB	DMKRND	DMKRNH	DMKRPA	DMKRSE	DMKRSP
		DMKSAV	DMKSCH	DMKSEP	DMKSEV	DMKSIX	DMKSNC	DMKSPL	DMKSSP	DMKSSS	DMKSTK	DMKSVC	DMKTAP
		DMKTCS	DMKTDK	DMKTHI	DMKTMR	DMKTRA	DMKTRC	DMKTRD	DMKTRK	DMKTRM	DMKUDR	DMKUDU	DMKUNT
		DMKUSO	DMKVAT	DMKVCA	DMKVCH	DMKVCN	DMKVDA	DMKVDC	DMKVDD	DMKVDE	DMKVDR	DMKVDS	DMKVER
		DMKVIO	DMKVMA	DMKVMC	DMKVMI	DMKVSI	DMKVSP	DMKVSQ	DMKWRM				
R13	000753	DMKACO	DMKALG	DMKAPI	DMKATS	DMKBLD	DMKBSC	DMKCCH	DMKCCW	DMKCDB	DMKCDM	DMKCDS	DMKCFC
		DMKCFD	DMKCFG	DMKCFH	DMKCFM	DMKCFO	DMKCFP	DMKCFS	DMKCFT	DMKCKP	DMKCKS	DMKCLK	DMKCNS
		DMKCPB	DMKCPI	DMKCPS	DMKCPU	DMKCPV	DMKCQG	DMKCQH	DMKCQP	DMKCQR	DMKCQY	DMKCSB	DMKCSO
		DMKCSP	DMKCSQ	DMKCST	DMKCSU	DMKCSV	DMKDAS	DMKDDR	DMKDEF	DMKDGD	DMKDIA	DMKDIB	DMKDIR
		DMKDMP	DMKDRD	DMKDSB	DMKDSP	DMKEIG	DMKENT	DMKERM	DMKEXT	DMKFMT	DMKFRE	DMKGIO	DMKGRF
		DMKGRT	DMKHVC	DMKHVD	DMKIMG	DMKIOC	DMKIOE	DMKIOF	DMKIOG	DMKIOS	DMKISM	DMKJRL	DMKLD00E
		DMKLNK	DMKLOG	DMKLOH	DMKMCC	DMKMCD	DMKMCH	DMKMIA	DMKMID	DMKMNI	DMKMON	DMKMSG	DMKMSW
		DMKNEM	DMKNES	DMKNET	DMKNLD	DMKNLE	DMKPAG	DMKPGS	DMKPGT	DMKPRG	DMKPRV	DMKPTR	DMKQCN
		DMKRGA	DMKRGB	DMKRND	DMKRNH	DMKRPA	DMKRSE	DMKRSP	DMKSAV	DMKSCH	DMKSEP	DMKSEV	DMKSIX
		DMKSNC	DMKSPL	DMKSSP	DMKSSS	DMKSVC	DMKTAP	DMKTCS	DMKTDK	DMKTHI	DMKTMR	DMKTRA	DMKTRC
		DMKTRD	DMKTRK	DMKTRM	DMKUDR	DMKUDU	DMKUNT	DMKUSO	DMKVAT	DMKVCA	DMKVCH	DMKVCN	DMKVDA
		DMKVDC	DMKVDD	DMKVDE	DMKVDR	DMKVDS	DMKVER	DMKVMA	DMKVMC	DMKVMI	DMKVSI	DMKVSP	DMKWRM
R14	007292	DMKACO	DMKALG	DMKAPI	DMKATS	DMKBLD	DMKBSC	DMKCCH	DMKCCW	DMKCDB	DMKCDM	DMKCDS	DMKCFC
		DMKCFD	DMKCFG	DMKCFH	DMKCFM	DMKCFO	DMKCFP	DMKCFS	DMKCFT	DMKCKP	DMKCKS	DMKCLK	DMKCNS
		DMKCPB	DMKCPI	DMKCPS	DMKCPU	DMKCPV	DMKCQG	DMKCQH	DMKCQP	DMKCQR	DMKCQY	DMKCSB	DMKCSO
		DMKCSP	DMKCSQ	DMKCST	DMKCSU	DMKCSV	DMKCVT	DMKDAS	DMKDDR	DMKDEF	DMKDGD	DMKDIA	DMKDIB
		DMKDIR	DMKDMP	DMKDRD	DMKDSB	DMKDSP	DMKEIG	DMKENT	DMKERM	DMKEXT	DMKFMT	DMKFRE	DMKGIO
		DMKGRF	DMKGRT	DMKHVC	DMKHVD	DMKIMG	DMKIOE	DMKIOF	DMKIOG	DMKIOS	DMKISM	DMKJRL	DMKLD00E
		DMKLNK	DMKLOC	DMKLOG	DMKLOH	DMKLOK	DMKMCC	DMKMCD	DMKMCH	DMKMCT	DMKMIA	DMKMID	DMKMNI
		DMKMON	DMKMSG	DMKMSW	DMKNES	DMKNET	DMKNLD	DMKNLE	DMKNMT	DMKOPR	DMKPAG	DMKPGS	DMKPGT
		DMKPRG	DMKPRV	DMKPSA	DMKPTR	DMKQCN	DMKRGA	DMKRGB	DMKRND	DMKRNH	DMKRPA	DMKRSE	DMKRSP
		DMKSAV	DMKSCH	DMKSCN	DMKSEP	DMKSEV	DMKSIX	DMKSNC	DMKSPL	DMKSSP	DMKSSS	DMKSTK	DMKSVC
		DMKTAP	DMKTCS	DMKTDK	DMKTHI	DMKTMR	DMKTRA	DMKTRC	DMKTRD	DMKTRK	DMKUDR	DMKUDU	DMKUNT
		DMKUSO	DMKVAT	DMKVCA	DMKVCH	DMKVCN	DMKVDA	DMKVDC	DMKVDD	DMKVDE	DMKVDR	DMKVDS	DMKVER
		DMKVIO	DMKVMA	DMKVMC	DMKVMI	DMKVSI	DMKVSP	DMKVSQ	DMKWRM				
R15	011326	DMKACO	DMKALG	DMKAPI	DMKATS	DMKBLD	DMKBSC	DMKCCH	DMKCCW	DMKCDB	DMKCDM	DMKCDS	DMKCFC
		DMKCFD	DMKCFG	DMKCFH	DMKCFM	DMKCFO	DMKCFP	DMKCFS	DMKCFT	DMKCKP	DMKCKS	DMKCLK	DMKCNS
		DMKCPB	DMKCPI	DMKCPS	DMKCPU	DMKCPV	DMKCQG	DMKCQH	DMKCQP	DMKCQR	DMKCQY	DMKCSB	DMKCSO
		DMKCSP	DMKCSQ	DMKCST	DMKCSU	DMKCSV	DMKCVT	DMKDAS	DMKDDR	DMKDEF	DMKDGD	DMKDIA	DMKDIB

LABEL	COUNT	REFERENCES											
		DMKDIR	DMKDMP	DMKDRD	DMKDSB	DMKDSP	DMKEIG	DMKENT	DMKERM	DMKEXT	DMKFMT	DMKFRE	DMKGIO
		DMKGRF	DMKGRT	DMKHVC	DMKHVD	DMKIMG	DMKIOE	DMKIOF	DMKIOG	DMKIOS	DMKISM	DMKJRL	DMKLD00E
		DMKLNK	DMKLOC	DMKLOG	DMKLOH	DMKLOK	DMKMCC	DMKMCD	DMKMCH	DMKMCT	DMKMIA	DMKMID	DMKMNI
		DMKMON	DMKMSG	DMKMSW	DMKNEM	DMKNES	DMKNET	DMKNLD	DMKNLE	DMKNMT	DMKOPR	DMKPAG	DMKPGS
		DMKPGT	DMKPRG	DMKPRV	DMKPSA	DMKPTR	DMKQCN	DMKRGA	DMKRGB	DMKRND	DMKRNH	DMKRPA	DMKRSE
		DMKRSP	DMKSAV	DMKSCH	DMKSCN	DMKSEP	DMKSEV	DMKSIX	DMKSNC	DMKSPL	DMKSSP	DMKSSS	DMKSTK
		DMKSVC	DMKTAP	DMKTCS	DMKTDK	DMKTHI	DMKTMR	DMKTRA	DMKTRC	DMKTRD	DMKTRK	DMKUDR	DMKUDU
		DMKUNT	DMKUSO	DMKVAT	DMKVCA	DMKVCH	DMKVCN	DMKVDA	DMKVDC	DMKVDD	DMKVDE	DMKVDR	DMKVDS
		DMKVER	DMKVIO	DMKVMA	DMKVMC	DMKVMI	DMKVSI	DMKVSP	DMKVSQ	DMKWRM			
R2	009554	DMKACO	DMKALG	DMKAPI	DMKATS	DMKBLD	DMKBSC	DMKCCH	DMKCCW	DMKCDE	DMKCDM	DMKCDS	DMKCFC
		DMKCFD	DMKCFG	DMKCFH	DMKCFM	DMKCFO	DMKCFP	DMKCFS	DMKCFT	DMKCKP	DMKCKS	DMKCLK	DMKCNS
		DMKCPB	DMKCPI	DMKCPS	DMKCPU	DMKCPV	DMKCQG	DMKCQH	DMKCQP	DMKCQR	DMKCQY	DMKCSB	DMKCSO
		DMKCSP	DMKCSQ	DMKCST	DMKCSU	DMKCSV	DMKCVT	DMKDAS	DMKDDR	DMKDEF	DMKDGD	DMKDIA	DMKDIB
		DMKDIR	DMKDMP	DMKDRD	DMKDSB	DMKDSP	DMKEIG	DMKENT	DMKERM	DMKEXT	DMKFMT	DMKFRE	DMKGIO
		DMKGRF	DMKGRT	DMKHVC	DMKHVD	DMKIMG	DMKIOE	DMKIOF	DMKIOG	DMKIOS	DMKISM	DMKJRL	DMKLD00E
		DMKLNK	DMKLOC	DMKLOG	DMKLOH	DMKLOK	DMKMCC	DMKMCD	DMKMCH	DMKMCT	DMKMIA	DMKMID	DMKMNI
		DMKMON	DMKMSG	DMKMSW	DMKNEM	DMKNES	DMKNET	DMKNLD	DMKNLE	DMKNMT	DMKOPR	DMKPAG	DMKPGS
		DMKPGT	DMKPRG	DMKPRV	DMKPSA	DMKPTR	DMKQCN	DMKRGA	DMKRGB	DMKRND	DMKRNH	DMKRPA	DMKRSE
		DMKRSP	DMKSAV	DMKSCH	DMKSCN	DMKSEP	DMKSEV	DMKSIX	DMKSNC	DMKSPL	DMKSSP	DMKSSS	DMKSTK
		DMKSVC	DMKTAP	DMKTCS	DMKTDK	DMKTHI	DMKTMR	DMKTRA	DMKTRC	DMKTRD	DMKTRK	DMKTRM	DMKUDR
		DMKUDU	DMKUNT	DMKUSO	DMKVAT	DMKVCA	DMKVCH	DMKVCN	DMKVDA	DMKVDC	DMKVDD	DMKVDE	DMKVDR
		DMKVDS	DMKVER	DMKVIO	DMKVMA	DMKVMC	DMKVMI	DMKVSI	DMKVSP	DMKVSQ	DMKWRM		
R3	006180	DMKACO	DMKALG	DMKAPI	DMKATS	DMKBLD	DMKBSC	DMKCCH	DMKCCW	DMKCDE	DMKCDM	DMKCDS	DMKCFC
		DMKCFD	DMKCFG	DMKCFH	DMKCFM	DMKCFO	DMKCFP	DMKCFS	DMKCFT	DMKCKP	DMKCKS	DMKCNS	DMKCPB
		DMKCPI	DMKCPS	DMKCPU	DMKCPV	DMKCQG	DMKCQH	DMKCQP	DMKCQR	DMKCQY	DMKCSB	DMKCSO	DMKCSP
		DMKCSQ	DMKCST	DMKCSU	DMKCSV	DMKCVT	DMKDAS	DMKDDR	DMKDEF	DMKDGD	DMKDIA	DMKDIR	DMKDMP
		DMKDRD	DMKDSB	DMKDSP	DMKEIG	DMKENT	DMKERM	DMKEXT	DMKFMT	DMKFRE	DMKGIO	DMKGRF	DMKGRT
		DMKHVC	DMKHVD	DMKIMG	DMKIOE	DMKIOF	DMKIOG	DMKIOS	DMKISM	DMKJRL	DMKLD00E	DMKLNK	DMKLOC
		DMKLOG	DMKLOH	DMKLOK	DMKMCC	DMKMCD	DMKMCH	DMKMCT	DMKMIA	DMKMID	DMKMNI	DMKMON	DMKMSG
		DMKMSW	DMKNEM	DMKNES	DMKNET	DMKNLD	DMKNLE	DMKNMT	DMKOPR	DMKPAG	DMKPGS	DMKPGT	DMKPRG
		DMKPRV	DMKPSA	DMKPTR	DMKQCN	DMKRGA	DMKRGB	DMKRND	DMKRNH	DMKRPA	DMKRSE	DMKRSP	DMKSAV
		DMKSCH	DMKSCN	DMKSEP	DMKSEV	DMKSIX	DMKSNC	DMKSPL	DMKSSP	DMKSSS	DMKSTK	DMKSVC	DMKTAP
		DMKTCS	DMKTDK	DMKTHI	DMKTMR	DMKTRA	DMKTRC	DMKTRD	DMKTRK	DMKTRM	DMKUDR	DMKUDU	DMKUNT
		DMKUSO	DMKVAT	DMKVCA	DMKVCH	DMKVCN	DMKVDA	DMKVDC	DMKVDD	DMKVDE	DMKVDR	DMKVDS	DMKVER
		DMKVIO	DMKVMA	DMKVMC	DMKVMI	DMKVSI	DMKVSP	DMKVSQ	DMKWRM				
R4	004873	DMKACO	DMKALG	DMKAPI	DMKATS	DMKBLD	DMKBSC	DMKCCH	DMKCCW	DMKCDE	DMKCDM	DMKCDS	DMKCFC
		DMKCFD	DMKCFG	DMKCFH	DMKCFM	DMKCFO	DMKCFP	DMKCFS	DMKCFT	DMKCKP	DMKCKS	DMKCNS	DMKCPB
		DMKCPI	DMKCPS	DMKCPU	DMKCPV	DMKCQG	DMKCQH	DMKCQP	DMKCQR	DMKCQY	DMKCSB	DMKCSO	DMKCSP
		DMKCSQ	DMKCST	DMKCSU	DMKCSV	DMKDAS	DMKDDR	DMKDEF	DMKDGD	DMKDIA	DMKDIB	DMKDIR	DMKDMP
		DMKDRD	DMKDSB	DMKDSP	DMKEIG	DMKENT	DMKERM	DMKEXT	DMKFMT	DMKFRE	DMKGIO	DMKGRF	DMKGRT
		DMKHVC	DMKHVD	DMKIMG	DMKIOC	DMKIOE	DMKIOF	DMKIOG	DMKIOS	DMKISM	DMKJRL	DMKLD00E	DMKLNK
		DMKLOC	DMKLOG	DMKLOH	DMKLOK	DMKMCC	DMKMCD	DMKMCH	DMKMCT	DMKMIA	DMKMID	DMKMNI	DMKMON
		DMKMSG	DMKMSW	DMKNEM	DMKNES	DMKNET	DMKNLD	DMKNLE	DMKNMT	DMKOPR	DMKPAG	DMKPGS	DMKPGT
		DMKPRG	DMKPRV	DMKPSA	DMKPTR	DMKQCN	DMKRGA	DMKRGB	DMKRND	DMKRNH	DMKRSE	DMKRSP	DMKSAV
		DMKSCH	DMKSCN	DMKSEP	DMKSEV	DMKSIX	DMKSNC	DMKSPL	DMKSSP	DMKSSS	DMKSTK	DMKSVC	DMKTAP
		DMKTCS	DMKTDK	DMKTHI	DMKTMR	DMKTRA	DMKTRC	DMKTRD	DMKTRK	DMKTRM	DMKUDR	DMKUDU	DMKUNT

LABEL	COUNT	REFERENCES											
		DMKUSO	DMKVAT	DMKVCA	DMKVCH	DMKVCN	DMKVDA	DMKVDC	DMKVDD	DMKVDE	DMKVDR	DMKVDS	DMKVER
		DMKVIO	DMKVMA	DMKVMC	DMKVMI	DMKVSI	DMKVSP	DMKVSQ	DMKWRM				
R5	004606	DMKACO	DMKATS	DMKBLD	DMKBSC	DMKCCH	DMKCCW	DMKCDB	DMKCDM	DMKCDS	DMKCFC	DMKCFD	DMKCFG
		DMKCFH	DMKCFM	DMKCFO	DMKCFP	DMKCFS	DMKCFT	DMKCKP	DMKCKS	DMKCNS	DMKCPB	DMKCPI	DMKCPS
		DMKCPU	DMKCPV	DMKCQG	DMKCQH	DMKCQP	DMKCQR	DMKCQY	DMKCSB	DMKCSO	DMKCSP	DMKCSQ	DMKCST
		DMKCSU	DMKCSV	DMKCVT	DMKDAS	DMKDDR	DMKDEF	DMKDGD	DMKDIA	DMKDIR	DMKDMP	DMKDRD	DMKDSB
		DMKDSP	DMKENT	DMKERM	DMKEXT	DMKFMT	DMKFRE	DMKGIO	DMKGRF	DMKGRT	DMKHVC	DMKHVD	DMKIMG
		DMKIOC	DMKIOE	DMKIOF	DMKIOG	DMKIOS	DMKISM	DMKJRL	DMKLD00E	DMKLNK	DMKLOC	DMKLOG	DMKLOH
		DMKLOK	DMKMCC	DMKMCD	DMKMCH	DMKMIA	DMKMID	DMKMNI	DMKMCN	DMKMSG	DMKMSW	DMKNEM	DMKNES
		DMKNET	DMKNLD	DMKNLE	DMKNMT	DMKOPR	DMKPAG	DMKPGS	DMKPGT	DMKPRG	DMKPRV	DMKPSA	DMKPTR
		DMKQCN	DMKRGA	DMKRGB	DMKRNH	DMKRPA	DMKRSE	DMKRSP	DMKSAV	DMKSCH	DMKSCN	DMKSEP	DMKSNC
		DMKSPL	DMKSSP	DMKSSS	DMKSTK	DMKSVC	DMKTAP	DMKTCS	DMKTDK	DMKTHI	DMKTMR	DMKTRA	DMKTRC
		DMKTRD	DMKTRK	DMKTRM	DMKUDR	DMKUDU	DMKUNT	DMKUSO	DMKVAT	DMKVCA	DMKVCH	DMKVCN	DMKVDA
		DMKVDC	DMKVDD	DMKVDE	DMKVDS	DMKVER	DMKVIC	DMKVMA	DMKVMC	DMKVMI	DMKVSI	DMKVSP	DMKVSQ
		DMKWRM											
R6	003264	DMKACO	DMKATS	DMKBLD	DMKBSC	DMKCCH	DMKCCW	DMKCDB	DMKCDM	DMKCDS	DMKCFC	DMKCFD	DMKCFG
		DMKCFH	DMKCFM	DMKCFO	DMKCFP	DMKCFS	DMKCFT	DMKCKP	DMKCKS	DMKCNS	DMKCPB	DMKCPI	DMKCPS
		DMKCPU	DMKCPV	DMKCQG	DMKCQP	DMKCQR	DMKCQY	DMKCSB	DMKCSO	DMKCSP	DMKCSQ	DMKCST	DMKCSU
		DMKCSV	DMKCVT	DMKDAS	DMKDDR	DMKDEF	DMKDGD	DMKDIA	DMKDIB	DMKDIR	DMKDMP	DMKDRD	DMKDSB
		DMKDSP	DMKENT	DMKERM	DMKEXT	DMKFMT	DMKFRE	DMKGIO	DMKGRF	DMKGRT	DMKHVC	DMKHVD	DMKIMG
		DMKIOC	DMKIOE	DMKIOF	DMKIOG	DMKIOS	DMKISM	DMKJRL	DMKLD00E	DMKLNK	DMKLOC	DMKLOG	DMKLOH
		DMKLOK	DMKMCC	DMKMCD	DMKMCH	DMKMCT	DMKMIA	DMKMID	DMKMNI	DMKMON	DMKMSW	DMKNES	DMKNET
		DMKNLD	DMKNLE	DMKNMT	DMKPAG	DMKPGT	DMKPRG	DMKPRV	DMKPTR	DMKQCN	DMKRGA	DMKRGB	DMKRND
		DMKRNH	DMKRSE	DMKRSP	DMKSAV	DMKSCH	DMKSCN	DMKSEP	DMKSNC	DMKSPL	DMKSSP	DMKSSS	DMKSTK
		DMKSVC	DMKTAP	DMKTCS	DMKTDK	DMKTHI	DMKTMR	DMKTRC	DMKTRD	DMKTRK	DMKUDR	DMKUDU	DMKUNT
		DMKUSO	DMKVAT	DMKVCA	DMKVCH	DMKVCN	DMKVDA	DMKVDC	DMKVDD	DMKVDE	DMKVDR	DMKVDS	DMKVER
		DMKVIO	DMKVMA	DMKVMC	DMKVMI	DMKVSI	DMKVSP	DMKVSQ	DMKWRM				
R7	003705	DMKACO	DMKALG	DMKATS	DMKBLD	DMKBSC	DMKCCH	DMKCCW	DMKCDB	DMKCDM	DMKCDS	DMKCFC	DMKCFD
		DMKCFG	DMKCFH	DMKCFM	DMKCFO	DMKCFP	DMKCFS	DMKCFT	DMKCKP	DMKCKS	DMKCNS	DMKCPB	DMKCPI
		DMKCPS	DMKCPU	DMKCPV	DMKCQG	DMKCQP	DMKCQR	DMKCQY	DMKCSO	DMKCSP	DMKCSQ	DMKCST	DMKCSU
		DMKCSV	DMKCVT	DMKDAS	DMKDDR	DMKDEF	DMKDGD	DMKDIA	DMKDIB	DMKDIR	DMKDMP	DMKDRD	DMKDSB
		DMKDSP	DMKENT	DMKERM	DMKFMT	DMKFRE	DMKGIC	DMKGRF	DMKGRT	DMKHVC	DMKHVD	DMKIMG	DMKIOE
		DMKIOF	DMKIOG	DMKIOS	DMKISM	DMKJRL	DMKLD00E	DMKLNK	DMKLOC	DMKLOG	DMKLOH	DMKMCC	DMKMCD
		DMKMCH	DMKMIA	DMKMID	DMKMNI	DMKMON	DMKMSG	DMKMSW	DMKNES	DMKNET	DMKNLD	DMKNLE	DMKNMT
		DMKPAG	DMKPGS	DMKPGT	DMKPRG	DMKPRV	DMKPSA	DMKPTR	DMKQCN	DMKRGA	DMKRGB	DMKRND	DMKRNH
		DMKRPA	DMKRSE	DMKRSP	DMKSAV	DMKSCH	DMKSCN	DMKSEP	DMKSNC	DMKSPL	DMKSSP	DMKSSS	DMKSVC
		DMKTAP	DMKTCS	DMKTDK	DMKTHI	DMKTMR	DMKTRC	DMKTRD	DMKTRK	DMKUDR	DMKUDU	DMKUNT	DMKUSO
		DMKVAT	DMKVCA	DMKVCH	DMKVCN	DMKVDA	DMKVDC	DMKVDD	DMKVDE	DMKVDS	DMKVER	DMKVIO	DMKVMA
		DMKVMC	DMKVMI	DMKVSI	DMKVSP	DMKVSQ	DMKWRM						
R8	002748	DMKACO	DMKALG	DMKATS	DMKBLD	DMKBSC	DMKCCH	DMKCCW	DMKCDB	DMKCDM	DMKCDS	DMKCFC	DMKCFD
		DMKCFG	DMKCFH	DMKCFM	DMKCFO	DMKCFP	DMKCFS	DMKCFT	DMKCKP	DMKCKS	DMKCNS	DMKCPB	DMKCPI
		DMKCPS	DMKCPU	DMKCPV	DMKCQG	DMKCQP	DMKCQR	DMKCQY	DMKCSB	DMKCSO	DMKCSP	DMKCSQ	DMKCST
		DMKCSU	DMKCSV	DMKCVT	DMKDAS	DMKDDR	DMKDEF	DMKDGD	DMKDIA	DMKDIB	DMKDMP	DMKDRD	DMKDSB
		DMKDSP	DMKENT	DMKERM	DMKFMT	DMKFRE	DMKGIC	DMKGRF	DMKGRT	DMKHVC	DMKHVD	DMKIMG	DMKIOC
		DMKIOE	DMKIOF	DMKIOG	DMKIOS	DMKISM	DMKJRL	DMKLD00E	DMKLNK	DMKLOC	DMKLOG	DMKLOH	DMKMCC
		DMKMCD	DMKMCH	DMKMIA	DMKMID	DMKMNI	DMKMON	DMKMSG	DMKMSW	DMKNES	DMKNET	DMKNLD	DMKNLE

LABEL	COUNT	REFERENCES											
		DMKNMT	DMKOPR	DMKPAG	DMKPGS	DMKPGT	DMKPRG	DMKPRV	DMKPSA	DMKPTR	DMKQCN	DMKRGA	DMKRGB
		DMKRNH	DMKRSE	DMKRSP	DMKSAV	DMKSCH	DMKSCN	DMKSEP	DMKSNC	DMKSPL	DMKSSP	DMKSSS	DMKSVC
		DMKTAP	DMKTCS	DMKTDK	DMKTHI	DMKTMR	DMKTRC	DMKTRD	DMKTRK	DMKTRM	DMKUDR	DMKUDU	DMKUNT
		DMKUSO	DMKVAT	DMKVCA	DMKVCH	DMKVCN	DMKVDA	DMKVDC	DMKVDD	DMKVDE	DMKVDR	DMKVDS	DMKVER
		DMKVIO	DMKVMA	DMKVMC	DMKVSI	DMKVSP	DMKVSQ	DMKWRM					
R9	002530	DMKACO	DMKALG	DMKATS	DMKBLD	DMKBSC	DMKCCH	DMKCCW	DMKCDM	DMKCDS	DMKCFC	DMKCFG	DMKCFH
		DMKCFM	DMKCFO	DMKCFP	DMKCFS	DMKCKP	DMKCKS	DMKCNS	DMKCPI	DMKCPS	DMKCPU	DMKCPV	DMKCQG
		DMKCQH	DMKCQP	DMKCQR	DMKCQY	DMKCSB	DMKCSC	DMKCSP	DMKCSQ	DMKCST	DMKCSU	DMKCSV	DMKCVT
		DMKDAS	DMKDDR	DMKDEF	DMKDGD	DMKDIA	DMKDIB	DMKDIR	DMKDMP	DMKDRD	DMKDSP	DMKEIG	DMKENT
		DMKERM	DMKFMT	DMKFRE	DMKGIO	DMKGRF	DMKHVC	DMKHVD	DMKIMG	DMKIOC	DMKIOE	DMKIOF	DMKIOG
		DMKIOS	DMKISM	DMKJRL	DMKLD00E	DMKLNK	DMKLOC	DMKLOG	DMKLOH	DMKLOK	DMKMCC	DMKMCD	DMKMCH
		DMKMCT	DMKMIA	DMKMID	DMKMNI	DMKMON	DMKMSG	DMKMSW	DMKNES	DMKNET	DMKNLD	DMKNLE	DMKNMT
		DMKPAG	DMKPGS	DMKPGT	DMKPRG	DMKPRV	DMKPTR	DMKQCN	DMKRGA	DMKRGE	DMKRNH	DMKRPA	DMKRSE
		DMKRSP	DMKSAV	DMKSCH	DMKSCN	DMKSEP	DMKSEV	DMKSIX	DMKSNC	DMKSPL	DMKSSP	DMKSSS	DMKTAP
		DMKTCS	DMKTDK	DMKTHI	DMKTMR	DMKTRA	DMKTRC	DMKTRD	DMKTRK	DMKUDR	DMKUDU	DMKUNT	DMKUSO
		DMKVAT	DMKVCA	DMKVCH	DMKVCN	DMKVDA	DMKVDC	DMKVDD	DMKVDE	DMKVDS	DMKVER	DMKVMA	DMKVMC
		DMKVMI	DMKVSI	DMKVSP	DMKVSQ	DMKWRM							
SAVCREGS	000007	DMKCFG	DMKCFH										
SAVE	000026	DMKBLD	DMKDSP	DMKEMB	DMKFRE	DMKIOS	DMKPRV	DMKPTR	DMKSCH	DMKTMR	DMKVMA	DMKVSI	
SAVEAR	000007	DMKIMG	DMKRND										
SAVEAREA	000175	DMKACO	DMKALG	DMKAPI	DMKATS	DMKBLD	DMKBSC	DMKCCH	DMKCCW	DMKCDB	DMKCDM	DMKCDS	DMKCFC
		DMKCFD	DMKCFG	DMKCFH	DMKCFM	DMKCFO	DMKCFP	DMKCFS	DMKCFT	DMKCKS	DMKCLK	DMKCNS	DMKCPB
		DMKCPS	DMKCPU	DMKCPV	DMKCQG	DMKCQH	DMKCQP	DMKCQR	DMKCQY	DMKCSB	DMKCSO	DMKCSP	DMKCSQ
		DMKCST	DMKCSU	DMKCSV	DMKDAS	DMKDDR	DMKDEF	DMKDGD	DMKDIA	DMKDIB	DMKDIR	DMKDRD	DMKDSB
		DMKEIG	DMKENT	DMKERM	DMKFMT	DMKGIO	DMKGRF	DMKGRT	DMKHVD	DMKIOC	DMKIOE	DMKIOF	DMKIOG
		DMKIOS	DMKISM	DMKJRL	DMKLNK	DMKLOG	DMKLOH	DMKMCC	DMKMCD	DMKMCH	DMKMIA	DMKMID	DMKMNI
		DMKMON	DMKMSG	DMKMSW	DMKNEM	DMKNES	DMKNET	DMKNLD	DMKNLE	DMKPGS	DMKPTR	DMKQCN	DMKRGA
		DMKRGB	DMKRNH	DMKRPA	DMKRSE	DMKRSP	DMKSEP	DMKSEV	DMKSIX	DMKSNC	DMKSPL	DMKSSP	DMKSSS
		DMKSVC	DMKTAP	DMKTCS	DMKTDK	DMKTHI	DMKTRA	DMKTRC	DMKTRD	DMKTRK	DMKTRM	DMKUDR	DMKUDU
		DMKUNT	DMKUSO	DMKVAT	DMKVCA	DMKVCH	DMKVDA	DMKVDC	DMKVDD	DMKVDE	DMKVDR	DMKVDS	DMKVER
		DMKVMA	DMKVMC	DMKVSP	DMKWRM								
SAVENEXT	000004	DMKSVC											
SAVEPROC	000010	DMKPTR	DMKSVC										
SAVEREGS	000545	DMKACO	DMKALG	DMKAPI	DMKATS	DMKBLD	DMKBSC	DMKCCH	DMKCCW	DMKCDB	DMKCDM	DMKCDS	DMKCFC
		DMKCFD	DMKCFG	DMKCFH	DMKCFM	DMKCFO	DMKCFP	DMKCFS	DMKCFT	DMKCKS	DMKCLK	DMKCPB	DMKCPS
		DMKCPU	DMKCPV	DMKCQG	DMKCQH	DMKCQP	DMKCQR	DMKCQY	DMKCSB	DMKCSO	DMKCSP	DMKCSQ	DMKCST
		DMKCSU	DMKCSV	DMKDAS	DMKDDR	DMKDEF	DMKDGD	DMKDIA	DMKDIB	DMKDIR	DMKDRD	DMKDSB	DMKENT
		DMKERM	DMKGIO	DMKGRT	DMKHVD	DMKIOC	DMKIOE	DMKIOF	DMKIOG	DMKIOS	DMKISM	DMKJRL	DMKLNK
		DMKLOG	DMKLOH	DMKMCC	DMKMCD	DMKMCH	DMKMIA	DMKMID	DMKMNI	DMKMON	DMKMSG	DMKMSW	DMKNEM
		DMKNES	DMKNET	DMKNLD	DMKNLE	DMKPGS	DMKPTR	DMKQCN	DMKRGB	DMKRNH	DMKRPA	DMKRSE	DMKRSP
		DMKSEP	DMKSNC	DMKSPL	DMKSSS	DMKTAP	DMKTCS	DMKTDK	DMKTHI	DMKTRA	DMKTRC	DMKTRD	DMKTRK
		DMKTRM	DMKUDR	DMKUDU	DMKUNT	DMKUSO	DMKVAT	DMKVCA	DMKVCH	DMKVDA	DMKVDC	DMKVDD	DMKVDE
		DMKVDR	DMKVDS	DMKVER	DMKVMC	DMKVMI	DMKVSP	DMKWRM					
SAVERETN	000034	DMKCFC	DMKCFG	DMKCPB	DMKDIA	DMKLNK	DMKLOG	DMKPTR	DMKSSS	DMKSVC	DMKUSO	DMKVAT	DMKVCA
		DMKVER											
SAVERTN	000001	DMKSVC											

LABEL	COUNT	REFERENCES											
SAVER0	000062	DMKCNS	DMKCQG	DMKCQH	DMKCQP	DMKCQR	DMKCQY	DMKDRD	DMKERM	DMKGRF	DMKHVD	DMKJRL	DMKLNK
		DMKMSW	DMKNEM	DMKPTR	DMKQCN	DMKRNH	DMKRSE	DMKSNC	DMKSVC	DMKTDK	DMKTRC	DMKTRD	DMKUDR
		DMKVAT	DMKVCA	DMKVDC	DMKVSP								
SAVER1	000080	DMKBLD	DMKCCW	DMKCFC	DMKCKS	DMKCNS	DMKCPU	DMKCQP	DMKERM	DMKIOE	DMKLNK	DMKMIA	DMKMON
		DMKPGS	DMKPTR	DMKQCN	DMKRNH	DMKRPA	DMKSSS	DMKTDK	DMKTRC	DMKTRD	DMKTRK	DMKVAT	DMKVCA
		DMKVDS	DMKVSP										
SAVER10	000019	DMKCCW	DMKSEP	DMKVCA	DMKVCH	DMKVDC	DMKVDE						
SAVER11	000079	DMKACO	DMKALG	DMKAPI	DMKBLD	DMKCFM	DMKCFC	DMKCFP	DMKCPS	DMKCPU	DMKCPV	DMKCQG	DMKCQP
		DMKCSV	DMKDAS	DMKDIA	DMKIOE	DMKIOS	DMKLNK	DMKLOG	DMKMID	DMKMSG	DMKMSW	DMKNES	DMKNLD
		DMKPTR	DMKQCN	DMKSPL	DMKTCS	DMKTHI	DMKUNT	DMKVCA	DMKVCH	DMKVDA	DMKVDD	DMKVMC	
SAVER12	000024	DMKCCW	DMKCPV	DMKDGD	DMKMCC	DMKMCD	DMKPGS	DMKPTR	DMKSSS	DMKSVC	DMKVAT	DMKVCA	DMKVER
SAVER13	000010	DMKPGS	DMKPTR	DMKSVC	DMKVAT	DMKVDC	DMKVDE						
SAVER2	000196	DMKACO	DMKALG	DMKBLD	DMKCCW	DMKCDS	DMKCFC	DMKCFM	DMKCKS	DMKCNS	DMKCPS	DMKCQG	DMKCQH
		DMKCQP	DMKDEF	DMKDGD	DMKDIA	DMKDIB	DMKDRD	DMKERM	DMKGIO	DMKGRF	DMKLNK	DMKLOG	DMKLOH
		DMKMIA	DMKMNI	DMKMSG	DMKNES	DMKNET	DMKNLD	DMKNLE	DMKPGS	DMKPTR	DMKQCN	DMKRGA	DMKRGB
		DMKRNH	DMKRPA	DMKSNC	DMKSSS	DMKTRA	DMKTRC	DMKTRD	DMKUDR	DMKVAT	DMKVCH	DMKVDA	DMKVDC
		DMKVDD	DMKVDE	DMKVDS	DMKVMC	DMKVSP	DMKWRM						
SAVER3	000011	DMKERM	DMKPGS	DMKPTR	DMKQCN	DMKVAT							
SAVER4	000002	DMKCCH	DMKTRC										
SAVER5	000010	DMKCFG	DMKMON	DMKTRC	DMKVCA	DMKVMC							
SAVER6	000009	DMKCFG	DMKDRD	DMKSNC	DMKVMC								
SAVER7	000003	DMKPTR	DMKSPL	DMKTRK									
SAVER8	000026	DMKBLD	DMKCCW	DMKCFP	DMKCKS	DMKDIA	DMKSEP	DMKSPL	DMKTDK	DMKTRK	DMKUNT	DMKVDR	DMKVDS
		DMKVSP											
SAVER9	000006	DMKALG	DMKCCW	DMKNES	DMKNET	DMKSPL							
SAVESIZE	000013	DMKCPI	DMKFRE	DMKLNK	DMKSVC	DMKVDA							
SAVEWRK1	001344	DMKACO	DMKALG	DMKATS	DMKBLD	DMKCCH	DMKCCW	DMKCDB	DMKCDM	DMKCDS	DMKCFC	DMKCFD	DMKCFG
		DMKCFH	DMKCFO	DMKCFS	DMKCFT	DMKCKS	DMKCPS	DMKCPU	DMKCPV	DMKCQG	DMKCQH	DMKCQP	DMKCQR
		DMKCQY	DMKCSB	DMKCSO	DMKCSP	DMKCSQ	DMKCST	DMKCSU	DMKCSV	DMKDEF	DMKDIA	DMKDIB	DMKDRD
		DMKEIG	DMKLNK	DMKLOG	DMKLOH	DMKMCC	DMKMCD	DMKMIA	DMKMON	DMKMSG	DMKNES	DMKNET	DMKNLD
		DMKNLE	DMKPGS	DMKPTR	DMKQCN	DMKRPA	DMKRSE	DMKSEP	DMKSEV	DMKSIX	DMKSNC	DMKSPL	DMKSSS
		DMKTAP	DMKTCS	DMKTHI	DMKTRA	DMKTRC	DMKTRD	DMKTRK	DMKUNT	DMKUSO	DMKVCA	DMKVCH	DMKVDA
		DMKVDC	DMKVDD	DMKVDE	DMKVDS	DMKVER	DMKVMC	DMKWRM					
SAVEWRK2	000696	DMKACO	DMKATS	DMKBLD	DMKCCH	DMKCCW	DMKCDM	DMKCDS	DMKCFC	DMKCFD	DMKCFG	DMKCFH	DMKCFO
		DMKCFS	DMKCKP	DMKCKS	DMKCPB	DMKCPS	DMKCPU	DMKCPV	DMKCQG	DMKCQH	DMKCQP	DMKCQR	DMKCQY
		DMKCSB	DMKCSO	DMKCSP	DMKCSQ	DMKCST	DMKCSU	DMKCSV	DMKDAS	DMKDEF	DMKDIA	DMKDRD	DMKIOG
		DMKIOS	DMKLNK	DMKLOG	DMKMID	DMKMSG	DMKNES	DMKNET	DMKNLD	DMKNLE	DMKPGS	DMKPTR	DMKQCN
		DMKRPA	DMKRSE	DMKSEP	DMKSNC	DMKSPL	DMKSSS	DMKSVC	DMKTCS	DMKTDK	DMKTHI	DMKTRA	DMKTRC
		DMKTRD	DMKTRK	DMKUDR	DMKUNT	DMKUSO	DMKVCA	DMKVCH	DMKVDA	DMKVDC	DMKVDD	DMKVDR	DMKVDS
		DMKVER	DMKVSP	DMKWRM									
SAVEWRK3	000200	DMKACO	DMKATS	DMKCCH	DMKCDS	DMKCFD	DMKCFG	DMKCFO	DMKCKS	DMKCPS	DMKCPU	DMKCPV	DMKCQG
		DMKCQP	DMKCSO	DMKDAS	DMKDEF	DMKDIA	DMKDRD	DMKMCC	DMKMCD	DMKMIA	DMKMSG	DMKNES	DMKNET
		DMKNLD	DMKNLE	DMKPGS	DMKQCN	DMKRSE	DMKSNC	DMKSSS	DMKTCS	DMKTHI	DMKTRC	DMKTRK	DMKUNT
		DMKUSO	DMKVCA	DMKVDA	DMKVDC	DMKVDD	DMKVDR	DMKVDS	DMKVER	DMKWRM			
SAVEWRK4	000253	DMKATS	DMKCCH	DMKCCW	DMKCDB	DMKCDM	DMKCDS	DMKCFC	DMKCFD	DMKCFG	DMKCFH	DMKCFO	DMKCFS
		DMKCKS	DMKCPB	DMKCPS	DMKCPU	DMKCPV	DMKCQG	DMKCQP	DMKCQY	DMKCSB	DMKCSO	DMKCSP	DMKCSQ

LABEL	COUNT	REFERENCES											
		DMKCST	DMKCSU	DMKCSV	DMKDEF	DMKDIA	DMKDRD	DMKLNK	DMKMCD	DMKMSG	DMKNES	DMKNET	DMKNLD
		DMKNLE	DMKPGS	DMKQCN	DMKRSE	DMKSNC	DMKSSS	DMKTAP	DMKTCS	DMKTHI	DMKTRC	DMKTRD	DMKTRK
		DMKUSO	DMKVCA	DMKVCH	DMKVDD	DMKVDR	DMKVDS	DMKVER	DMKWRM				
SAVEWRK5	000248	DMKATS	DMKCCH	DMKCDB	DMKCDM	DMKCDS	DMKCFD	DMKCFG	DMKCFH	DMKCFO	DMKCFS	DMKCKS	DMKCPB
		DMKCPU	DMKCQG	DMKCQH	DMKCQP	DMKCQY	DMKCSC	DMKCSP	DMKCSQ	DMKCST	DMKCSU	DMKCSV	DMKDEF
		DMKDIA	DMKDRD	DMKLNK	DMKNES	DMKNET	DMKNLD	DMKNLE	DMKPGS	DMKSEP	DMKSNC	DMKSSS	DMKTCS
		DMKTRC	DMKTRK	DMKUNT	DMKVCH	DMKVDA	DMKVDC	DMKVDD	DMKVER	DMKVMC			
SAVEWRK6	000242	DMKACO	DMKATS	DMKCCH	DMKCDB	DMKCDM	DMKCFG	DMKCFH	DMKCFM	DMKCFP	DMKCFS	DMKCKS	DMKCPS
		DMKCPU	DMKCQG	DMKCQP	DMKCQY	DMKCSB	DMKCSC	DMKCSP	DMKCSQ	DMKCST	DMKCSU	DMKCSV	DMKDEF
		DMKDIA	DMKDRD	DMKLNK	DMKLOG	DMKMCD	DMKMSG	DMKNLD	DMKNLE	DMKPGS	DMKPTR	DMKRSE	DMKSNC
		DMKSSS	DMKTCS	DMKTHI	DMKTRC	DMKTRD	DMKTRK	DMKUNT	DMKVCA	DMKVCH	DMKVDA	DMKVDC	DMKVDD
		DMKVDR	DMKVDS	DMKVMC	DMKVSP	DMKWRM							
SAVEWRK7	000145	DMKACO	DMKCCH	DMKCFD	DMKCFG	DMKCFH	DMKCFC	DMKCFS	DMKCKS	DMKCPS	DMKCPU	DMKCQG	DMKCQH
		DMKCQP	DMKCQY	DMKCSB	DMKCSO	DMKCSP	DMKCST	DMKCSU	DMKCSV	DMKDEF	DMKDIA	DMKIOG	DMKLNK
		DMKNES	DMKNET	DMKNLD	DMKNLE	DMKPGS	DMKSEP	DMKSSS	DMKTCS	DMKTRA	DMKTRC	DMKTRD	DMKTRK
		DMKVDA	DMKVDC	DMKVDD	DMKVDS	DMKVER	DMKVMC	DMKWRM					
SAVEWRK8	000304	DMKACO	DMKALG	DMKATS	DMKCCH	DMKCDB	DMKCDM	DMKCDS	DMKCFO	DMKCFP	DMKCFS	DMKCKS	DMKCPS
		DMKCPU	DMKCPV	DMKCQG	DMKCQH	DMKCQP	DMKCQR	DMKCSB	DMKCSO	DMKCSP	DMKCSQ	DMKCST	DMKCSU
		DMKCSV	DMKDEF	DMKDIA	DMKLNK	DMKLOG	DMKLOH	DMKMSG	DMKNES	DMKNET	DMKNLD	DMKNLE	DMKPTR
		DMKSEP	DMKSNC	DMKSPL	DMKSSS	DMKTCS	DMKTRC	DMKTRD	DMKTRK	DMKVCH	DMKVDA	DMKVDC	DMKVDD
		DMKVER	DMKWRM										
SAVEWRK9	000188	DMKACO	DMKATS	DMKBLD	DMKCCH	DMKCCW	DMKCFG	DMKCFH	DMKCKS	DMKCPS	DMKCPU	DMKCQY	DMKCSP
		DMKCSQ	DMKCSV	DMKDEF	DMKDGD	DMKDIA	DMKEIG	DMKLNK	DMKLOG	DMKLOH	DMKNES	DMKNET	DMKNLD
		DMKNLE	DMKPGS	DMKPTR	DMKSEP	DMKSEV	DMKSIX	DMKSNC	DMKTCS	DMKTRC	DMKTRD	DMKUNT	DMKVCA
		DMKVCH	DMKVDA	DMKVDC	DMKVDD	DMKVDR	DMKVDS	DMKVER					
SAVFPRES	000002	DMKCFG	DMKCFH										
SAVGREGS	000003	DMKCFG	DMKCFH										
SAVKEYS	000002	DMKCFG	DMKCFH										
SAVPSW	000002	DMKCFG	DMKCFH										
SAVTABLE	000002	DMKCFG	DMKCFH										
SBA	000115	DMKDDR	DMKDIR	DMKFMT	DMKGRF	DMKGRT	DMKGRW	DMKOPR	DMKRGA	DMKRGB	DMKSSP		
SCHEDCL	000001	DMKMCC											
SDRBLOK	000012	DMKIOE	DMKIOF										
SDRBSIZE	000001	DMKIOE											
SDRCTRS	000014	DMKIOE	DMKIOF										
SDRCUA	000002	DMKIOF											
SDRFLAGS	000005	DMKIOE	DMKIOF										
SDRFLCT	000004	DMKIOF											
SDRLNGTH	000009	DMKIOE	DMKIOF										
SDRMAX	000003	DMKIOF											
SDROVFWK	000006	DMKIOF											
SDRPRMCT	000004	DMKIOF											
SDRRDEV	000002	DMKIOF											
SDRRECD	000004	DMKIOF											
SDRSHRT	000007	DMKIOE	DMKIOF										
SDRSIZE	000001	DMKIOF											

LABEL	COUNT	REFERENCES											
SDRSIZE1	000001	DMKIOP											
SEARCH	000002	DMKCFC	DMKVMI										
SEEK	000020	DMKCKP	DMKDAS	DMKDMP	DMKSPL	DMKTDK	DMKVMI						
SEEKADR	000006	DMKDAS											
SEEKCL	000001	DMKIOS											
SEGENQ	000002	DMKBLD											
SEGINV	000028	DMKATS	DMKBLD	DMKCFG	DMKCFH	DMKPGS	DMKPTR	DMKVMA					
SEGPAGE	000072	DMKATS	DMKBLD	DMKCFG	DMKCPI	DMKPGS	DMKPTR	DMKVMA					
SEGPLEN	000006	DMKBLD	DMKVMA										
SEGTABLE	000014	DMKATS	DMKBLD	DMKPGS	DMKVMA								
SENSE	000077	DMKDDR	DMKDIR	DMKDMP	DMKEXT	DMKFMT	DMKSSP	DMKVMI	DMKVSP				
SETUP	000011	DMKACO	DMKCKP	DMKCPI	DMKDSP								
SETUP1	000002	DMKACO	DMKCKP										
SETUP2	000002	DMKACO	DMKCKP										
SF	000097	DMKDDR	DMKDIR	DMKFMT	DMKGRF	DMKGRT	DMKGRW	DMKOPR	DMKRGA	DMKRGB	DMKSSP		
SFBCLAS	000030	DMKCKP	DMKCQH	DMKCQR	DMKCSP	DMKCSQ	DMKCSU	DMKCSV	DMKDRD	DMKMIA	DMKNLE	DMKRSP	DMKSEP
		DMKSPL	DMKVSP	DMKVSQ									
SFBCOPY	000027	DMKCKP	DMKCKS	DMKCQH	DMKCSO	DMKCSU	DMKDRD	DMKMIA	DMKNLE	DMKRSP	DMKSPL	DMKTCS	
SFBDATE	000013	DMKCKP	DMKCQH	DMKDMP	DMKMIA	DMKNLE	DMKSEP	DMKSPL	DMKWRM				
SFBDIST	000035	DMKCKP	DMKCKS	DMKCQH	DMKCSQ	DMKCSU	DMKMIA	DMKNLE	DMKSEP	DMKSPL			
SFBDUMP	000008	DMKCKS	DMKCQH	DMKDMP	DMKDRD	DMKNLE	DMKSPL	DMKVSP					
SFBEOF	000014	DMKCKS	DMKDRD	DMKVSP	DMKWRM								
SFBFILID	000046	DMKCKP	DMKCKS	DMKCQH	DMKCSP	DMKCSQ	DMKCST	DMKCSU	DMKCSV	DMKDMP	DMKDRD	DMKMIA	DMKMNI
		DMKNLE	DMKRSE	DMKRSP	DMKSEP	DMKSPL	DMKTCS	DMKVSP	DMKVSQ	DMKWRM			
SFBFIRST	000002	DMKCKP	DMKSPL										
SFBFLAG	000101	DMKCKP	DMKCKS	DMKCQH	DMKCQR	DMKCSO	DMKCSP	DMKCSQ	DMKCSU	DMKCSV	DMKDRD	DMKNLE	DMKRSE
		DMKRSP	DMKSPL	DMKUSO	DMKVSP	DMKVSQ	DMKWRM						
SFBFLAG2	000047	DMKCKP	DMKCKS	DMKCSO	DMKCSQ	DMKDRD	DMKMIA	DMKRSP	DMKSEP	DMKSPL	DMKVSP	DMKVSQ	DMKWRM
SFBFLASH	000007	DMKCQH	DMKCSU	DMKRSP	DMKSPL								
SFBFLNMT	000002	DMKVSP	DMKVSQ										
SFBFNAME	000015	DMKCKP	DMKCQH	DMKCSQ	DMKCSU	DMKMIA	DMKNLE	DMKRSP	DMKSEP	DMKSPL	DMKWRM		
SFBFTYPE	000009	DMKCKP	DMKCQH	DMKMIA	DMKNLE	DMKRSP	DMKWRM						
SFBHOLD	000011	DMKCSQ	DMKSPL	DMKVSP									
SFBINUSE	000018	DMKCKS	DMKCQH	DMKCQR	DMKCSP	DMKCSQ	DMKCSU	DMKCSV	DMKDRD	DMKRSP	DMKUSO	DMKVSP	DMKWRM
SFBLAST	000055	DMKCKP	DMKCKS	DMKDMP	DMKDRD	DMKMIA	DMKNLE	DMKRSP	DMKSPL	DMKVSP	DMKVSQ		
SFBLOK	000085	DMKCKP	DMKCKS	DMKCPI	DMKCQH	DMKCQR	DMKCSC	DMKCSP	DMKCSQ	DMKCST	DMKCSU	DMKCSV	DMKDMP
		DMKDRD	DMKMIA	DMKMNI	DMKNLE	DMKRSE	DMKRSP	DMKSEP	DMKSPL	DMKTCS	DMKUSO	DMKVSP	DMKVSQ
		DMKWRM											
SFBMISC1	000003	DMKVSQ											
SFBMON	000007	DMKCKS	DMKDRD	DMKMIA	DMKSPL	DMKVSP							
SFBNOHLD	000010	DMKCSQ	DMKSPL	DMKVSP									
SFBOPEN	000007	DMKCKS	DMKDRD	DMKVSP	DMKWRM								
SFBORIG	000013	DMKCKP	DMKCPI	DMKCQH	DMKCSV	DMKMIA	DMKNLE	DMKRSP	DMKSEP	DMKSPL			
SFBPNT	000083	DMKCKP	DMKCKS	DMKCQH	DMKCQR	DMKCST	DMKCSV	DMKDMP	DMKDRD	DMKMIA	DMKNLE	DMKRSE	DMKRSP
		DMKSPL	DMKVSP	DMKVSQ	DMKWRM								
SFBPURGE	000011	DMKCKP	DMKSPL	DMKVSP	DMKVSQ								

LABEL	COUNT	REFERENCES											
SFBRECER	000033	DMKCKP	DMKCKS	DMKCSO	DMKDRD	DMKRSE	DMKRSP	DMKSPL	DMKVSP	DMKVSQ	DMKWRM		
SFBRECNO	000028	DMKCKS	DMKCQH	DMKMIA	DMKNLE	DMKRSP	DMKSEP	DMKSPL	DMKVSP	DMKVSQ			
SFBRECOK	000003	DMKCSO	DMKRSP										
SFBRECS	000019	DMKCKP	DMKCKS	DMKRSP	DMKSPL	DMKVSP	DMKWRM						
SFBRECSZ	000006	DMKNLE	DMKSPL	DMKVSP									
SFBREQUE	000007	DMKCSO	DMKRSP	DMKSPL									
SFBRSTRT	000008	DMKCKS	DMKRSP	DMKSEP	DMKSPL	DMKWRM							
SFBSHOLD	000019	DMKCQH	DMKCQR	DMKCSO	DMKCSQ	DMKCSU	DMKRSE	DMKRSP	DMKSPL				
SFBSIZE	000040	DMKCKP	DMKCKS	DMKDMP	DMKDRD	DMKMIA	DMKNLE	DMKRSP	DMKSPL	DMKUSO	DMKWRM		
SFBSTART	000081	DMKCKP	DMKCKS	DMKCPI	DMKCQH	DMKCST	DMKCSU	DMKDMP	DMKDRD	DMKMIA	DMKNLE	DMKRSP	DMKSPL
		DMKTCS	DMKVSP	DMKVSQ									
SFBSTCPY	000004	DMKSPL	DMKTCS										
SFBTICER	000001	DMKRSP											
SFBTIME	000014	DMKCKS	DMKCQH	DMKDMP	DMKMIA	DMKNLE	DMKSEP	DMKSPL	DMKVSP	DMKVSQ			
SFBTYPE	000015	DMKCQH	DMKDMP	DMKDRD	DMKMIA	DMKNLE	DMKRSE	DMKSPL	DMKVSQ				
SFBUHOLD	000019	DMKCKS	DMKCQH	DMKCQR	DMKCSP	DMKCSU	DMKDRD	DMKRSP	DMKSPL	DMKVSP			
SFBUSER	000046	DMKCKP	DMKCPI	DMKCQH	DMKCQR	DMKCSP	DMKCSQ	DMKCST	DMKCSU	DMKCSV	DMKDRD	DMKMIA	DMKNLE
		DMKRSP	DMKSEP	DMKSPL	DMKTCS	DMKVSP							
SHQBLOK	000011	DMKCKS	DMKCQR	DMKCSQ	DMKSPL	DMKWRM							
SHQBSIZE	000012	DMKCKP	DMKCKS	DMKCSQ	DMKWRM								
SHQFLAGS	000001	DMKCSQ											
SHQPNT	000001	DMKCSQ											
SHQSHOLD	000005	DMKCQR	DMKCSQ	DMKSPL									
SHQUSER	000007	DMKCKS	DMKCQR	DMKCSQ	DMKSPL								
SHRBPNT	000007	DMKATS	DMKCFG	DMKCFH	DMKPGS								
SHRFLAG	000016	DMKATS	DMKCFG	DMKCPU	DMKPGS	DMKPTR							
SHRFPNT	000022	DMKATS	DMKCFG	DMKCFH	DMKCPU	DMKPGS							
SHRLKCNT	000012	DMKCCW	DMKDGD										
SHRNAME	000017	DMKATS	DMKCFG	DMKCFH	DMKCPU	DMKPGS	DMKVMA						
SHRNOPRT	000016	DMKATS	DMKCFG	DMKCPU	DMKPGS	DMKPTR							
SHRPAGE	000025	DMKATS	DMKCDS	DMKCFG	DMKCPU								
SHRSEGCT	000019	DMKATS	DMKCFG	DMKCPU	DMKPGS	DMKPTR	DMKVMA						
SHRSEGNM	000016	DMKATS	DMKCFG	DMKCPU	DMKPGS	DMKVMA							
SHRTABLE	000032	DMKATS	DMKCFG	DMKCFH	DMKCPU	DMKPGS	DMKPTR	DMKVMA					
SHRTSIZE	000003	DMKATS	DMKCFG	DMKPGS									
SHRUSECT	000009	DMKATS	DMKCFG	DMKPGS									
SIGAPR	000001	DMKMCT											
SIGCLK	000001	DMKCLK											
SIGDISP	000001	DMKDSP											
SIGEMS	000010	DMKACO	DMKCLK	DMKCPS	DMKCPU	DMKEXT	DMKPTR						
SIGEXT	000001	DMKPTR											
SIGIPR	000003	DMKCPI	DMKCPU										
SIGMASK	000001	DMKDSP											
SIGQUI	000005	DMKACO	DMKCLK	DMKCPS	DMKCPU	DMKPTR							
SIGRES	000004	DMKACO	DMKCLK	DMKPTR									
SIGREST	000004	DMKAPI	DMKDMP	DMKMCT									

LABEL	COUNT	REFERENCES											
SIGSAVE	000006	DMKEXT											
SIGSENSE	000005	DMKCPI	DMKCPU	DMKDMP	DMKEXT								
SIGSHD	000001	DMKCPS											
SIGSSS	000005	DMKDMP	DMKMCT										
SIGSTART	000002	DMKMCH											
SIGSTOP	000004	DMKCPU	DMKDMP	DMKMCH	DMKMCT								
SIGSYNC	000001	DMKCLK											
SIGWAKE	000010	DMKCPI	DMKCPU	DMKDSP	DMKFRE	DMKIOS	DMKSCH						
SIGXC	000017	DMKACO	DMKCLK	DMKCPI	DMKCPU	DMKDSP	DMKEXT	DMKFRE	DMKICS	DMKMCT	DMKPTR	DMKSCH	
SILI	001230	DMKACO	DMKBSC	DMKCCW	DMKCFP	DMKCKP	DMKCNS	DMKCPB	DMKCPI	DMKCSE	DMKDAS	DMKDDR	DMKDGD
		DMKDIB	DMKDIR	DMKDMP	DMKDSB	DMKFMT	DMKGRF	DMKGRT	DMKGRW	DMKIOS	DMKLD00E	DMKMCC	DMKNLD
		DMKNLE	DMKOPR	DMKPAG	DMKRGA	DMKRGB	DMKRNH	DMKRSE	DMKRSP	DMKSAV	DMKSEP	DMKSPL	DMKSSP
		DMKTAP	DMKTCS	DMKTDK	DMKTRK	DMKUCB	DMKUCC	DMKUCS	DMKUDR	DMKVCA	DMKVCN	DMKVDE	DMKVDR
		DMKVMI	DMKVSP	DMKVSQ	DMKWRM								
SIOCCH	000002	DMKCCH											
SIZE	000024	DMKCKP	DMKFRE	DMKLD00E									
SKIP	000162	DMKACO	DMKCCW	DMKCKP	DMKCNS	DMKCSB	DMKCST	DMKDAS	DMKDDR	DMKDIB	DMKDIR	DMKDMP	DMKDRD
		DMKFMT	DMKIOS	DMKPAG	DMKRSE	DMKRSP	DMKSAV	DMKSEP	DMKSPL	DMKTAP	DMKTCS	DMKTRK	DMKUNT
		DMKVCA	DMKVCN	DMKVDE	DMKVMI	DMKVSI	DMKVSP						
SM	000031	DMKCNS	DMKCPI	DMKDMP	DMKFMT	DMKIOS	DMKNLD	DMKNLE	DMKPSA	DMKRSE	DMKSSP	DMKVMI	DMKVSI
SPCHAR	000012	DMKCQH	DMKCSU	DMKSPL	DMKTCS								
SPCMOD	000008	DMKCQH	DMKCSU	DMKSPL	DMKTCS								
SPCOPYFG	000007	DMKCQH	DMKCSU	DMKSPL	DMKTCS								
SPEC	000082	DMKLD00E											
SPENDSIZ	000002	DMKSPL											
SPFCB	000008	DMKCQH	DMKCSU	DMKSPL	DMKTCS								
SPFILID	000004	DMKCKS	DMKVSP	DMKVSQ									
SPFLAG1	000005	DMKCQH	DMKCSU	DMKSPL	DMKTCS								
SPFLSHC	000005	DMKCQH	DMKCSU	DMKSPL	DMKTCS								
SPLINK	000015	DMKCKS	DMKCQH	DMKCSU	DMKDRD	DMKMIA	DMKRSP	DMKSPL	DMKTCS	DMKVSP	DMKVSQ		
SPNXTPAG	000029	DMKCKS	DMKDRD	DMKMIA	DMKRSP	DMKVSP	DMKVSQ						
SPOOLED	000035	DMKCPS	DMKDMP	DMKMCC	DMKMCD	DMKMIA	DMKMNI	DMKMON					
SPPREPAG	000018	DMKCKS	DMKDRD	DMKMIA	DMKRSP	DMKVSP	DMKVSQ						
SPRECNUM	000017	DMKCKS	DMKMIA	DMKRSP	DMKVSP	DMKVSQ							
SPRMISC	000002	DMKRSP											
SPROFCL	000003	DMKMCC	DMKMON										
SPSIZE	000014	DMKRSP	DMKSPL	DMKVSP	DMKVSQ								
SPTIME	000004	DMKCKS	DMKVSP	DMKVSQ									
SSAVE	000004	DMKIMG											
STACKVM	000012	DMKCPU	DMKDSP	DMKUSO									
START	000037	DMKACO	DMKAPI	DMKBLD	DMKCFC	DMKCPU	DMKDIA	DMKDSP	DMKEME	DMKFMT	DMKIOS	DMKLD00E	DMKLOG
		DMKMCD	DMKMCH	DMKMCT	DMKPRG	DMKPSA	DMKSAV	DMKSCH	DMKSVC	DMKUSO			
STARTIME	000013	DMKBLD	DMKCKP	DMKCPI	DMKCQR	DMKWRM							
STCODE	000004	DMKACO	DMKCKP										
STOP	000026	DMKACO	DMKBLD	DMKCPU	DMKDSP	DMKEXT	DMKMCH	DMKMCT	DMKMNI	DMKPRG	DMKSCH	DMKSVC	
STRTADDR	000001	DMKIMG											

LABEL	COUNT	REFERENCES											
SUSPEND	000015	DMKCFP	DMKCKP	DMKMIA	DMKMNI	DMKMON							
SVCNPSW	000014	DMKAPI	DMKCPI	DMKPRG	DMKSVC	DMKTRC							
SVCOPSW	000037	DMKPRG	DMKSVC	DMKTRC									
SVCREGS	000013	DMKSVC											
SVMNOUPD	000002	DMKCFO	DMKLOK										
SVMSTAY	000017	DMKCNS	DMKGRF	DMKLOK	DMKNES	DMKNLD	DMKRGA	DMKRGB	DMKRNH	DMKSPL			
SVMUNLOK	000007	DMKCNS	DMKCPV	DMKLOK	DMKRGA	DMKRNH	DMKSPL						
SWPALLOC	000013	DMKATS	DMKPGS	DMKPTR									
SWPAPP	000005	DMKATS	DMKCFG	DMKPTR									
SWPCHG1	000012	DMKATS	DMKCFG	DMKCPI	DMKCPU	DMKMCH	DMKPTR	DMKRPA					
SWPCHG2	000006	DMKMCH	DMKPTR	DMKRPA									
SWPCODE	000007	DMKATS	DMKCPU	DMKPAG	DMKPGS	DMKPTR							
SWPCYL	000016	DMKATS	DMKCDS	DMKCFG	DMKCPI	DMKCPU	DMKPAG	DMKPGS	DMKPGT	DMKPTR	DMKRPA	DMKUDU	
SWPDPAGE	000008	DMKPAG	DMKPGT	DMKPTR									
SWPFLAG	000133	DMKATS	DMKBLD	DMKCCW	DMKCDS	DMKCFG	DMKCFH	DMKCPI	DMKCPU	DMKDGD	DMKMCH	DMKPAG	DMKPGS
		DMKPGT	DMKPRV	DMKPTR	DMKRPA	DMKUDU	DMKVAT	DMKVMA					
SWPFLAG2	000005	DMKATS	DMKCFG	DMKPTR									
SWPKEY1	000018	DMKATS	DMKCFG	DMKCFH	DMKMCH	DMKPGS	DMKPRV	DMKPTR					
SWPKEY2	000006	DMKMCH	DMKPTR										
SWPPAG	000004	DMKATS	DMKBLD	DMKCFG									
SWPRECMP	000026	DMKATS	DMKBLD	DMKCDS	DMKCPU	DMKPAG	DMKPGS	DMKPGT	DMKPTR	DMKRPA	DMKUDU		
SWPREF1	000004	DMKPTR											
SWPREF2	000005	DMKPTR											
SWPSHR	000012	DMKATS	DMKCFG	DMKPGS	DMKPRV	DMKPTR	DMKRPA	DMKVAT					
SWPTABLE	000040	DMKATS	DMKBLD	DMKCFG	DMKCPU	DMKPGS	DMKPTR	DMKVAT	DMKVMA				
SWPTRANS	000016	DMKCDS	DMKPAG	DMKPGS	DMKPTR	DMKRPA	DMKVMA						
SWPVM	000016	DMKATS	DMKBLD	DMKCFG	DMKCPU	DMKPGS							
SWPVPAGE	000009	DMKATS	DMKPTR	DMKVMA									
SWTCH	000013	DMKMCH	DMKMCT										
SWTHSAVE	000010	DMKSTK	DMKVMA										
SYNCLOG	000001	DMKCPI											
SYNCMASK	000002	DMKCLK	DMKEXT										
SYSCYL	000002	DMKCFG	DMKCFH										
SYSFLAG	000003	DMKCFG											
SYSHRSEG	000006	DMKCFG	DMKCPU										
SYSIPLDV	000009	DMKCKS	DMKCPI	DMKHVD	DMKIOG	DMKWRM							
SYSLOCS	000021	DMKACO	DMKBLD	DMKCFO	DMKCFT	DMKCKP	DMKLOC	DMKLOG	DMKLOH	DMKUDR	DMKUDU	DMKUSO	
SYSNAME	000011	DMKATS	DMKCFG	DMKCFH	DMKCPU								
SYSPAGCT	000003	DMKCFG	DMKCFH										
SYSPAGLN	000012	DMKATS	DMKCFG	DMKCFH	DMKCPU								
SYSPAGNM	000022	DMKATS	DMKCFG	DMKCFH	DMKCPU								
SYSPNT	000005	DMKATS	DMKCFG	DMKCFH	DMKCPU								
SYSPROT	000003	DMKCFG											
SYSSEGLN	000004	DMKCFG											
SYSSIZE	000002	DMKCFG	DMKCFH										
SYSSTART	000004	DMKATS	DMKCFG	DMKCFH	DMKCPU								

LABEL	COUNT	REFERENCES											
SYSTBL	000005	DMKATS	DMKCFG	DMKCFH	DMKCPU								
SYSTEM	000318	DMKAPI	DMKATS	DMKCFC	DMKCFG	DMKCFH	DMKCKS	DMKCNS	DMKCPB	DMKCPI	DMKCPS	DMKCPU	DMKCPV
		DMKCQH	DMKCSB	DMKCSO	DMKCST	DMKCSU	DMKDRD	DMKEMC	DMKERM	DMKGRF	DMKGRT	DMKHVD	DMKIOF
		DMKIOG	DMKMCC	DMKMIA	DMKMON	DMKNLD	DMKNLE	DMKPTR	DMKRGA	DMKRGB	DMKRNH	DMKRPA	DMKRSP
		DMKSEP	DMKSNC	DMKSPL	DMKSSP	DMKSVC	DMKTCS	DMKUDR	DMKUDU	DMKVCH	DMKVDR	DMKVSP	DMKVSQ
		DMKWRM											
SYSTEMID	000006	DMKVMC											
SYSVADDR	000006	DMKCFG	DMKCFH										
SYSVIRT	000033	DMKCCW	DMKCFP	DMKCQP	DMKDEF	DMKDGD	DMKDSB	DMKIOS	DMKLNK	DMKLOG	DMKSSS	DMKVDA	DMKVSI
SYSVOL	000008	DMKATS	DMKCFG	DMKCFH	DMKCPU								
TABEND	000002	DMKRGA											
TAG	000003	DMKCST											
TAPE	000004	DMKCCW	DMKDDR	DMKVDC	DMKVMI								
TBLCT	000007	DMKLD00E											
TBLREF	000007	DMKLD00E											
TEMPR0	000059	DMKAPI	DMKATS	DMKCNS	DMKCPI	DMKDSP	DMKPTR	DMKRGA	DMKVCA	DMKVMA	DMKVSP		
TEMPR1	000014	DMKCQP	DMKPGS	DMKPTR	DMKSCH	DMKVSP							
TEMPR10	000002	DMKCCW											
TEMPR11	000002	DMKATS											
TEMPR12	000002	DMKPRG											
TEMPR14	000028	DMKCCW	DMKCDS	DMKCPI	DMKPRG	DMKSCH							
TEMPR15	000008	DMKCCW	DMKCDS	DMKCPI	DMKPRG	DMKSCH							
TEMPR2	000029	DMKAPI	DMKATS	DMKCCW	DMKCPI	DMKCSP	DMKCSU	DMKCSV	DMKPGS	DMKPTR	DMKSAV	DMKVCA	DMKVMA
TEMPR3	000016	DMKAPI	DMKCCW	DMKCPI	DMKCSU	DMKCSV	DMKRGB	DMKVCA					
TEMPR4	000011	DMKAPI	DMKCPI	DMKCSU	DMKCSV	DMKSAV	DMKSCH	DMKVCA					
TEMPR5	000008	DMKAPI	DMKCPI	DMKVCA	DMKVMA								
TEMPR6	000006	DMKHVC	DMKVMA										
TEMPR7	000002	DMKRGA											
TEMPR8	000002	DMKHVC											
TEMPR9	000002	DMKIOS											
TEMPSAVE	000173	DMKAPI	DMKCCW	DMKCFS	DMKCNS	DMKCPI	DMKCQR	DMKCVT	DMKDSP	DMKFRE	DMKGRF	DMKIOS	DMKLOG
		DMKMID	DMKNET	DMKNLD	DMKPRV	DMKPSA	DMKQCN	DMKRNH	DMKSAV	DMKSCH	DMKSSS	DMKVIO	DMKVSI
TEMPST	000007	DMKLD00E											
TERMSYS	000009	DMKCCH	DMKEIG	DMKSEV	DMKSIX								
TESTCON	000001	DMKDDR											
TEXT	000011	DMKIMG	DMKNMT	DMKRND									
TEXTA	000002	DMKDDR	DMKDIR										
TIC	000008	DMKSSP	DMKTAP	DMKTDK	DMKVCN	DMKVMI	DMKVSP						
TIMEDISP	000180	DMKACO	DMKALG	DMKAPI	DMKBLD	DMKCFO	DMKCFP	DMKCNS	DMKCPI	DMKCPS	DMKCPU	DMKCPV	DMKCSU
		DMKCSV	DMKDIA	DMKDSP	DMKEXT	DMKGRF	DMKIOE	DMKIOS	DMKJRL	DMKLOG	DMKLOH	DMKLOK	DMKMCH
		DMKMCT	DMKMIA	DMKMID	DMKMSG	DMKMSW	DMKNES	DMKNLD	DMKPAG	DMKPRG	DMKPSA	DMKPTR	DMKQCN
		DMKRGA	DMKRGB	DMKRNH	DMKSCH	DMKSPL	DMKSVC	DMKTCS	DMKUSO	DMKVCA	DMKVCH	DMKVDA	DMKVDD
		DMKVMC	DMKVSI										
TIMER	000036	DMKAPI	DMKCFS	DMKCPI	DMKDDR	DMKDIR	DMKDSP	DMKEMB	DMKEXT	DMKIOS	DMKMCH	DMKPRG	DMKPSA
		DMKPTR	DMKSCH	DMKSVC	DMKTRA								
TIOCCH	000012	DMKCCH	DMKEIG	DMKSEV	DMKSIX								

LABEL	COUNT	REFERENCES										
TMPLOC	000010	DMKLD00E										
TNSCPIDN	000001	DMKIOF										
TNSDEVAD	000016	DMKIOF										
TNSKEYN	000002	DMKIOF										
TNSREC	000003	DMKIOF										
TNSSNS1	000005	DMKIOF										
TNSSWS3	000010	DMKIOF										
TNSVOLID	000002	DMKIOF										
TODATE	000022	DMKACO	DMKCKP	DMKCPI	DMKCVT	DMKDDR	DMKDMP	DMKMCD	DMKMID	DMKMNI		
TODSYNC	000001	DMKCLK										
TOOBIG	000001	DMKMCD										
TRACBEF	000003	DMKCNS	DMKIOS	DMKVSI								
TRACCURR	000065	DMKCNS	DMKCPI	DMKDSP	DMKEXT	DMKFRE	DMKIOS	DMKLOK	DMKMCC	DMKMCH	DMKPRG	DMKPSA	DMKRNH
		DMKSCH	DMKSVC	DMKVSI									
TRACEFLG	000008	DMKCPI	DMKMCC	DMKMCD	DMKMIA								
TRACEND	000022	DMKCNS	DMKCPI	DMKDSP	DMKEXT	DMKFRE	DMKIOS	DMKLOK	DMKMCH	DMKPRG	DMKPSA	DMKRNH	DMKSCH
		DMKSVC	DMKVSI										
TRACFLG1	000011	DMKFRE	DMKIOS	DMKMCH	DMKPRG	DMKPSA	DMKSCH	DMKSVC					
TRACFLG2	000014	DMKCNS	DMKDSP	DMKEXT	DMKIOS	DMKLOK	DMKRNH	DMKVSI					
TRACPROC	000023	DMKAPI	DMKCNS	DMKCPI	DMKDSP	DMKEXT	DMKFRE	DMKIOS	DMKLCK	DMKMCH	DMKMON	DMKPRG	DMKPSA
		DMKRNH	DMKSCH	DMKSVC	DMKVSI								
TRACSTRT	000027	DMKAPI	DMKCNS	DMKCPI	DMKCPU	DMKDSP	DMKEXT	DMKFRE	DMKIOS	DMKLOK	DMKMCC	DMKMCH	DMKPRG
		DMKPSA	DMKRNH	DMKSCH	DMKSVC	DMKVSI							
TRAC0A	000001	DMKDSP											
TRAC0C	000001	DMKDSP											
TRAC0D	000001	DMKVSI											
TRAC01	000001	DMKPSA											
TRAC02	000002	DMKSVC											
TRAC03	000002	DMKPRG											
TRAC04	000001	DMKMCH											
TRAC05	000001	DMKIOS											
TRAC08	000001	DMKSCH											
TRAC09	000001	DMKSCH											
TRAC10	000001	DMKDSP											
TRAC11	000001	DMKRNH											
TRAC12	000001	DMKLOK											
TRAC13	000005	DMKEXT											
TRAC67	000002	DMKFRE											
TRANMODE	000024	DMKCDS	DMKCFG	DMKCPB	DMKDSP	DMKMCH	DMKPRG	DMKPRV	DMKSVC	DMKTMR	DMKTRC	DMKTRD	DMKVAT
		DMKVER											
TRAP	000004	DMKMIA	DMKMON										
TRCCLCH	000003	DMKVSI											
TRCCSW	000003	DMKVSI											
TRCDROP	000003	DMKSCH											
TRCEXT	000003	DMKPSA											
TRCFREE	000003	DMKFRE											

LABEL	COUNT	REFERENCES										
TRCFRET	000003	DMKFRE										
TRCHALT	000003	DMKVSI										
TRCLOK	000003	DMKLOK										
TRCMCH	000003	DMKMCH										
TRCNCP	000003	DMKRNH										
TRCPGM	000006	DMKPRG										
TRCRUN	000001	DMKDSP										
TRCSCH	000003	DMKSCH										
TRCSIGP	000003	DMKEXT										
TRCSVC	000006	DMKSVC										
TRCUNBLK	000001	DMKDSP										
TRCUNSTK	000001	DMKDSP										
TREXADD	000004	DMKPRG	DMKVAT									
TREXANSI	000006	DMKPGS	DMKTRA	DMKTRC	DMKTRD							
TREXBRAN	000013	DMKTRA	DMKTRC	DMKTRD								
TREXBUFF	000008	DMKTRC	DMKTRD									
TREXCCW	000006	DMKTRA	DMKTRD									
TREXCR9	000004	DMKDSP	DMKPRV	DMKTMR								
TREXCSW	000006	DMKTRA	DMKTRC	DMKVIO								
TREXCTL	000004	DMKTRA										
TREXCTL1	000003	DMKTRC	DMKTRD									
TREXCTL2	000009	DMKTRC	DMKTRD	DMKVIO								
TREXFLAG	000012	DMKDSP	DMKPRV	DMKTRC	DMKTRD							
TREXINST	000013	DMKTRA	DMKTRC	DMKTRD								
TREXINTC	000004	DMKPRG	DMKPRV									
TREXINTL	000001	DMKPRG										
TREXIN1	000012	DMKDSP	DMKPGS	DMKPRV	DMKSVC	DMKTRA	DMKTRC	DMKTRD				
TREXIN2	000009	DMKDSP	DMKTRC	DMKTRD								
TREXLCNT	000006	DMKTRC	DMKTRD									
TREXLOCK	000001	DMKCFM										
TREXNDSP	000008	DMKDSP	DMKPRV	DMKTRC	DMKTRD							
TREXNSI	000012	DMKPGS	DMKPRV	DMKTRC								
TREXPERA	000003	DMKPRG	DMKPRV	DMKTMR								
TREXPERC	000001	DMKPRG										
TREXPRNT	000005	DMKTRA	DMKTRC	DMKTRD								
TREXPSW	000002	DMKPRG										
TREXRUNF	000006	DMKTRA	DMKTRC	DMKTRD								
TREXSIZE	000006	DMKTRA	DMKTRC	DMKTRD	DMKUSO							
TREXSVC1	000003	DMKTRC	DMKTRD									
TREXSVC2	000003	DMKTRC	DMKTRD									
TREXT	000023	DMKCFM	DMKDSP	DMKPGS	DMKPRG	DMKPRV	DMKSVC	DMKTMR	DMKTRA	DMKTRC	DMKTRD	DMKVIO
TREXTERM	000008	DMKCFM	DMKTRA	DMKTRC	DMKTRD							
TREXVAT	000005	DMKTRC	DMKTRD									
TRQBBPNT	000007	DMKPSA	DMKSCH									
TRQBFPNT	000025	DMKCFP	DMKCPU	DMKDIA	DMKDSP	DMKPSA	DMKSCH	DMKTMR	DMKUSO			
TRQBIRA	000014	DMKBLD	DMKCFC	DMKCFS	DMKCPI	DMKGRF	DMKLOG	DMKMCC	DMKMNI	DMKQCN	DMKRGA	DMKSSS

LABEL	COUNT	REFERENCES											
TRQBLOK	000080	DMKBLD	DMKCDS	DMKCFC	DMKCFM	DMKCFP	DMKCFS	DMKCPI	DMKCPU	DMKDIA	DMKDSP	DMKENT	DMKGRF
		DMKLOG	DMKMCC	DMKMID	DMKMNI	DMKMON	DMKPSA	DMKQCN	DMKRGA	DMKRGB	DMKSCH	DMKSSS	DMKTMR
		DMKUSO											
TRQBQUE	000018	DMKCFP	DMKSCH	DMKTMR									
TRQBSIZE	000042	DMKBLD	DMKCFC	DMKCFM	DMKCFS	DMKCPI	DMKDIA	DMKGRF	DMKLOG	DMKMCC	DMKMCD	DMKMNI	DMKQCN
		DMKRGA	DMKSSS	DMKUSO									
TRQBTOD	000036	DMKCFC	DMKCFM	DMKCPI	DMKENT	DMKMCC	DMKMNI	DMKMON	DMKSCH	DMKSSS	DMKTMR		
TRQBUSER	000015	DMKBLD	DMKCFC	DMKCFS	DMKCPI	DMKDSP	DMKGRF	DMKLOG	DMKMCC	DMKMNI	DMKQCN	DMKRGA	DMKSSS
TRQBVAL	000039	DMKCDS	DMKCFC	DMKCFM	DMKCFP	DMKCPI	DMKCPU	DMKDSP	DMKENT	DMKGRF	DMKMCC	DMKMID	DMKMNI
		DMKMON	DMKPSA	DMKQCN	DMKRGA	DMKSCH	DMKSSS	DMKTMR					
TRUN	000008	DMKDMP	DMKMNI	DMKMON									
TTSEGCNT	000008	DMKATS	DMKBLD	DMKCPU									
TYPBSC	000044	DMKACO	DMKBLD	DMKCFM	DMKCFT	DMKCKP	DMKCPI	DMKCPS	DMKCQG	DMKCQP	DMKCQR	DMKCQY	DMKDIA
		DMKHVD	DMKIOF	DMKIOS	DMKLOG	DMKLOH	DMKNES	DMKNLD	DMKQCN	DMKRGA	DMKUSO	DMKVCN	DMKVDE
		DMKVDS	DMKWRM										
TYPCTCA	000027	DMKCFP	DMKCPB	DMKCQG	DMKDEF	DMKDIA	DMKDIE	DMKDIR	DMKIOS	DMKSCN	DMKVCA	DMKVDA	DMKVDC
		DMKVDR	DMKVDS	DMKVMI	DMKVSI								
TYPE	000147	DMKAPI	DMKBLD	DMKCFC	DMKCKP	DMKCPU	DMKDDR	DMKDIR	DMKDSP	DMKEXT	DMKFMT	DMKFRE	DMKIOS
		DMKMCH	DMKMCT	DMKPAG	DMKPRG	DMKPRV	DMKPSA	DMKPTR	DMKSCH	DMKSTK	DMKSVC	DMKTMR	DMKVAT
		DMKVMA	DMKVMI	DMKVSI	DMKWRM								
TYPIBM1	000006	DMKDEF	DMKDIA	DMKDIR	DMKNLD	DMKSCN							
TYPPRT	000068	DMKCFO	DMKCKP	DMKCQG	DMKCQH	DMKCQR	DMKCSE	DMKCSO	DMKCSP	DMKCSQ	DMKCST	DMKCSU	DMKCSV
		DMKDEF	DMKDMP	DMKDRD	DMKNLE	DMKRSP	DMKSCN	DMKSPL	DMKSSP	DMKVSP	DMKVSQ		
TYPPUN	000060	DMKCKP	DMKCQR	DMKCSO	DMKCSP	DMKCSQ	DMKCST	DMKCSU	DMKCSV	DMKDRD	DMKRSE	DMKRSP	DMKSCN
		DMKSEP	DMKSPL	DMKSSP	DMKVSP								
TYPRDR	000019	DMKCQH	DMKCSP	DMKCSQ	DMKCST	DMKCSU	DMKCSV	DMKDRD	DMKRSP	DMKSCN	DMKSPL	DMKVMI	DMKVSI
TYPTELE2	000005	DMKDEF	DMKDIA	DMKDIR	DMKSCN								
TYPTIMER	000005	DMKCQG	DMKDIR	DMKVSP									
TYPTR	000001	DMKLD00E											
TYPTTY	000014	DMKCFT	DMKCNS	DMKCPV	DMKIOF	DMKNES							
TYPUNDEF	000003	DMKCNS	DMKNES	DMKNLD									
TYPUNSUP	000003	DMKCCW	DMKVMI										
TYP1050	000006	DMKCNS	DMKIOF										
TYP1052	000009	DMKDEF	DMKDIR	DMKLOG	DMKSPL	DMKVDR	DMKVDS	DMKVSP					
TYP1403	000006	DMKDEF	DMKDIR	DMKDMP	DMKIOF	DMKRSE							
TYP1442R	000001	DMKCCW											
TYP1443	000006	DMKDIR	DMKIOF	DMKRSE									
TYP2305	000074	DMKATS	DMKCCW	DMKCFH	DMKCKP	DMKCPI	DMKCPS	DMKCPU	DMKCQG	DMKCQP	DMKDAS	DMKDDR	DMKDEF
		DMKDGD	DMKDIR	DMKDRD	DMKDSB	DMKHVD	DMKIOC	DMKIOE	DMKIOF	DMKIOG	DMKIOS	DMKLOG	DMKPAG
		DMKPGS	DMKPGT	DMKPTR	DMKSAV	DMKUNT	DMKVDA	DMKVDC	DMKVDD	DMKVDE	DMKVDR	DMKVDS	DMKVER
		DMKWRM											
TYP2311	000019	DMKCCW	DMKCQG	DMKDDR	DMKDGD	DMKDIR	DMKIOC	DMKLNK	DMKSCN	DMKVDS			
TYP2314	000057	DMKATS	DMKCCW	DMKCFG	DMKCFH	DMKCKP	DMKCKS	DMKCPI	DMKCPU	DMKCQG	DMKDAS	DMKDDR	DMKDGD
		DMKDIR	DMKDMP	DMKDSB	DMKHVD	DMKLNK	DMKNLD	DMKNLE	DMKPAG	DMKPGT	DMKSAV	DMKSNC	DMKSPL
		DMKSSP	DMKTCS	DMKTDK	DMKVDC	DMKVER	DMKVSI						
TYP2319	000004	DMKDDR	DMKDRD	DMKHVD									

LABEL	COUNT	REFERENCES											
TYP2401	000006	DMKDDR	DMKTAP	DMKVMI									
TYP2415	000004	DMKDDR	DMKTAP	DMKVMI									
TYP2420	000004	DMKDDR	DMKTAP	DMKVMI									
TYP2501	000005	DMKDIR	DMKIOF	DMKRSE	DMKVMI								
TYP2520P	000002	DMKRSE											
TYP2520R	000001	DMKIOF											
TYP2540P	000005	DMKACO	DMKDIR	DMKMIA	DMKRSE	DMKSSP							
TYP2540R	000008	DMKDIR	DMKIOF	DMKRSE	DMKRSP	DMKSSP	DMKVMI						
TYP2700	000003	DMKIOF	DMKNES	DMKSCN									
TYP2741	000013	DMKCFC	DMKCNS	DMKCPI	DMKIOF								
TYP2955	000001	DMKCCW											
TYP3066	000013	DMKCPI	DMKCPV	DMKGRF	DMKIOF	DMKOPR	DMKSSP						
TYP3138	000002	DMKDIR											
TYP3148	000002	DMKDIR											
TYP3158	000002	DMKDIR											
TYP3203	000016	DMKCSB	DMKDEF	DMKDIR	DMKIOF	DMKRSE	DMKVSP						
TYP3210	000045	DMKCCW	DMKCFP	DMKCKP	DMKCNS	DMKCPB	DMKCPI	DMKCQG	DMKCSP	DMKCSQ	DMKCST	DMKDIA	DMKDIR
		DMKHVD	DMKIOF	DMKSCN	DMKSPL	DMKSSP	DMKVCN	DMKVDS	DMKVSI	DMKVSP	DMKVSQ		
TYP3211	000024	DMKCSB	DMKDEF	DMKDIR	DMKIOF	DMKRSE	DMKSPL	DMKVDR	DMKVSP				
TYP3215	000004	DMKCFP	DMKDIR										
TYP3277	000036	DMKACO	DMKCCW	DMKCFT	DMKCKP	DMKCPI	DMKCPV	DMKDIA	DMKDIB	DMKDIR	DMKGRF	DMKHVD	DMKSSP
		DMKVDS											
TYP3278	000026	DMKCCW	DMKCFT	DMKCKP	DMKCPI	DMKCPV	DMKGRF	DMKHVD	DMKVDS				
TYP3284	000014	DMKCKP	DMKCPV	DMKGRF									
TYP3330	000080	DMKATS	DMKCCW	DMKCFG	DMKCFH	DMKCFP	DMKCKP	DMKCKS	DMKCPI	DMKCPU	DMKCQG	DMKCQP	DMKDAS
		DMKDDR	DMKDGD	DMKDIR	DMKDMP	DMKDRD	DMKDSB	DMKFMT	DMKHVD	DMKIOC	DMKIOE	DMKIOF	DMKIOG
		DMKIOS	DMKLNK	DMKLOG	DMKNLD	DMKNLE	DMKPAG	DMKPGT	DMKSAV	DMKSNC	DMKSSS	DMKTCS	DMKTDK
		DMKUNT	DMKVDA	DMKVDC	DMKVER	DMKVSI	DMKWRM						
TYP3340	000070	DMKCCW	DMKCFH	DMKCKP	DMKCPI	DMKCQG	DMKDAS	DMKDDR	DMKDGD	DMKDIR	DMKDMP	DMKDRD	DMKDSB
		DMKGIO	DMKHVD	DMKIOE	DMKIOF	DMKIOG	DMKMSW	DMKPAG	DMKPGT	DMKSAV	DMKSPL	DMKSSP	DMKTDK
		DMKUNT	DMKVDC	DMKVDE	DMKVER	DMKVIO	DMKVSI	DMKWRM					
TYP3350	000076	DMKATS	DMKCCW	DMKCFG	DMKCFH	DMKCKP	DMKCKS	DMKCPI	DMKCPU	DMKCQG	DMKDAS	DMKDDR	DMKDGD
		DMKDIR	DMKDMP	DMKDRD	DMKFMT	DMKHVD	DMKIOE	DMKIOF	DMKIOG	DMKMSW	DMKNLD	DMKNLE	DMKPAG
		DMKPGT	DMKSAV	DMKSNC	DMKSPL	DMKSSP	DMKTCS	DMKTDK	DMKUNT	DMKVDC	DMKVER	DMKVSI	DMKWRM
TYP3410	000008	DMKCCW	DMKDDR	DMKIOE	DMKIOF	DMKTAP							
TYP3411	000002	DMKDDR											
TYP3420	000009	DMKCCW	DMKDDR	DMKDMP	DMKIOE	DMKIOF	DMKTAP						
TYP3505	000008	DMKDIR	DMKIOF	DMKRSE	DMKVSP								
TYP3525	000002	DMKDIR											
TYP3704	000002	DMKCCW	DMKVIO										
TYP3705	000019	DMKBLD	DMKCCW	DMKCKP	DMKCPS	DMKCQP	DMKNES	DMKNET	DMKNLD	DMKNLE	DMKRNH	DMKSCN	DMKUSO
		DMKVCH	DMKVDC	DMKVDS	DMKVSI	DMKWRM							
TYP3800	000036	DMKCCW	DMKCKP	DMKCKS	DMKCPI	DMKCPS	DMKCQP	DMKCSO	DMKIOF	DMKIOS	DMKRSE	DMKRSP	DMKSEP
		DMKSSP	DMKVDR	DMKVDS	DMKWRM								
TYP3851	000005	DMKCCW	DMKCFP	DMKCKP	DMKCPI	DMKSCN							
UC	000097	DMKBSC	DMKCKP	DMKCNS	DMKCPI	DMKDAS	DMKDDR	DMKDIB	DMKDIR	DMKDMP	DMKDSB	DMKDSP	DMKFMT

LABEL	COUNT	REFERENCES											
		DMKGRF	DMKHVC	DMKIOE	DMKIOS	DMKLDOOE	DMKMON	DMKNLD	DMKNLE	DMKOPR	DMKRNH	DMKRSE	DMKRSP
		DMKSSP	DMKTAP	DMKTRK	DMKUNT	DMKVCA	DMKVCN	DMKVDC	DMKVDE	DMKVIO	DMKVMI	DMKVSI	DMKVSP
UCASE	000032	DMKCFH	DMKCFM	DMKCFO	DMKCNS	DMKCPI	DMKGRF	DMKLNK	DMKMSW	DMKNLD	DMKNLE	DMKQCN	DMKRGA
		DMKRNH											
UCNTRL	000004	DMKUDU											
UCNTRLSZ	000003	DMKUDU											
UCURPASS	000002	DMKUDU											
UDASDDEV	000004	DMKUDU											
UDASDDIR	000002	DMKUDU											
UDASDMAC	000002	DMKUDU											
UDBFBLOK	000017	DMKCFS	DMKDEF	DMKHVD	DMKLNK	DMKLOG	DMKSPL	DMKUDR					
UDBFDASD	000004	DMKUDR											
UDBFSIZE	000016	DMKCFS	DMKDEF	DMKHVD	DMKLNK	DMKLOG	DMKLOH	DMKSPL					
UDBFVADD	000013	DMKCFS	DMKDEF	DMKHVD	DMKLNK	DMKLOG	DMKSPL	DMKUDR					
UDBFWORK	000006	DMKUDR											
UDEVAD	000002	DMKUDU											
UDEVADD	000048	DMKDEF	DMKDIR	DMKLNK	DMKLOG	DMKSSS	DMKUDR	DMKUDU	DMKVDA	DMKVDS			
UDEVBLOK	000051	DMKDEF	DMKDIR	DMKLNK	DMKLOG	DMKSCN	DMKSSS	DMKUDR	DMKUDU	DMKVDA	DMKVDS		
UDEVCLAS	000006	DMKDEF	DMKDIR	DMKVDS									
UDEVCODE	000001	DMKUDU											
UDEVDASD	000005	DMKDIR	DMKUDR	DMKUDU									
UDEVDED	000003	DMKDIR	DMKLNK	DMKLOG									
UDEVDISP	000011	DMKDEF	DMKDIR	DMKLNK	DMKLOG	DMKUDR	DMKUDU						
UDEVF	000001	DMKUDU											
UDEVFTR	000010	DMKDEF	DMKDIR	DMKLNK	DMKLOG	DMKSCN	DMKVDS						
UDEVLINK	000010	DMKDIR	DMKLNK	DMKLOG	DMKSSS								
UDEVLKDV	000004	DMKDIR	DMKLNK	DMKLOG									
UDEVLKID	000010	DMKDIR	DMKLNK	DMKLOG									
UDEVLM	000045	DMKDIR	DMKLNK	DMKUDU									
UDEVLONG	000006	DMKDIR	DMKLNK	DMKLOG	DMKUDU								
UDEVLR	000015	DMKDIR	DMKLNK	DMKUDU									
UDEVLW	000027	DMKDIR	DMKLNK	DMKUDU									
UDEVMODE	000017	DMKDIR	DMKLNK	DMKLOG	DMKUDU	DMKVDA	DMKVDS						
UDEVMR	000002	DMKDIR	DMKUDU										
UDEVMW	000002	DMKDIR	DMKUDU										
UDEVNCYL	000007	DMKDEF	DMKDIR	DMKLNK	DMKVDS								
UDEVPASM	000003	DMKDIR											
UDEVPASR	000006	DMKDIR	DMKLNK	DMKUDU									
UDEVPASW	000003	DMKDIR											
UDEVR	000002	DMKDIR	DMKLNK										
UDEVRELN	000004	DMKDIR	DMKLNK	DMKSCN	DMKVDS								
UDEVRR	000002	DMKDIR	DMKUDU										
UDEVSIZE	000018	DMKDIR	DMKLOG	DMKUDR									
UDEVSPOO	000001	DMKDIR											
UDEVSTAT	000026	DMKDEF	DMKDIR	DMKLNK	DMKLOG	DMKUDU	DMKVDS						
UDEVTDSK	000006	DMKDEF	DMKDIR	DMKLNK	DMKLOG	DMKVDS							

LABEL	COUNT	REFERENCES												
UDEVTYPC	000026	DMKDEF	DMKDIR	DMKLNK	DMKLOG	DMKVDS								
UDEVTYPE	000020	DMKDEF	DMKDIR	DMKLNK	DMKLOG	DMKVDS								
UDEVVRR	000004	DMKDIR	DMKUDU	DMKVDS										
UDEVVSER	000027	DMKDIR	DMKLNK	DMKLOG	DMKSSS									
UDEVW	000004	DMKDIR	DMKLNK	DMKVDA										
UDEVWR	000002	DMKDIR	DMKUDU											
UDEV3158	000003	DMKDEF	DMKDIR	DMKVDS										
UDIRAD	000002	DMKUDU												
UDIRBLOK	000021	DMKCFS	DMKCPI	DMKCSP	DMKDEF	DMKDIR	DMKHVD	DMKLNK	DMKLOG	DMKSPL	DMKUDR	DMKUDU		
UDIRDASD	000004	DMKDIR	DMKUDR	DMKUDU										
UDIRDISP	000010	DMKCFS	DMKDEF	DMKDIR	DMKHVD	DMKLNK	DMKLOG	DMKSPL	DMKUDR	DMKUDU				
UDIRF	000004	DMKUDU												
UDIRPASS	000010	DMKCPI	DMKCSP	DMKDIR	DMKLOG	DMKUDU								
UDIRSIZE	000011	DMKCSP	DMKDIR	DMKUDR	DMKUDU									
UDIRUSER	000013	DMKDIR	DMKHVD	DMKLOG	DMKUDR									
UDISPDEV	000003	DMKUDU												
UDISPMAC	000002	DMKUDU												
UE	000056	DMKCNS	DMKCSB	DMKDDR	DMKDIR	DMKDMP	DMKFMT	DMKGIO	DMKGRF	DMKHVC	DMKIOG	DMKIOS	DMKMON	
		DMKRGA	DMKRNH	DMKRSP	DMKSEP	DMKSSP	DMKVCA	DMKVCN	DMKVMI	DMKVSI	DMKVSP			
UFLAGS	000026	DMKUDU												
UIPARMS	000002	DMKUDU												
UIPARMSZ	000004	DMKUDU												
ULOCDVAD	000001	DMKUDU												
UMACACC	000002	DMKDIR	DMKLOG											
UMACACCT	000013	DMKDIR	DMKHVD	DMKLOG	DMKUDU									
UMACAD	000009	DMKUDU												
UMACAFF	000008	DMKCFS	DMKDIR	DMKLOG	DMKUDU									
UMACBLOK	000031	DMKCFS	DMKDEF	DMKDIR	DMKHVD	DMKLOG	DMKSPL	DMKUDR	DMKUDU					
UMACBMX	000002	DMKDIR	DMKLOG											
UMACCDEL	000002	DMKDIR	DMKLOG											
UMACCLA	000001	DMKLOG												
UMACCLEV	000006	DMKDIR	DMKLOG	DMKUDU										
UMACCORE	000003	DMKDIR	DMKLOG	DMKUDU										
UMACCPU	000002	DMKDIR	DMKLOG											
UMACDASD	000002	DMKDIR	DMKUDR											
UMACDISP	000003	DMKDIR	DMKUDR											
UMACDIST	000010	DMKDIR	DMKLOG	DMKSPL	DMKUDU									
UMACDVCT	000004	DMKDIR	DMKLOG	DMKUDR										
UMACECOP	000002	DMKDIR	DMKLOG											
UMACES	000002	DMKDIR	DMKLOG											
UMACF	000001	DMKUDU												
UMACFFON	000003	DMKDIR	DMKLOG	DMKUDU										
UMACIPL	000005	DMKDIR	DMKLOG	DMKUDU										
UMACISAM	000002	DMKDIR	DMKLOG											
UMACLDEL	000002	DMKDIR	DMKLOG											
UMACLEND	000005	DMKDIR	DMKLOG	DMKUDU										

LABEL	COUNT	REFERENCES											
UMACMCOR	000004	DMKDEF	DMKDIR	DMKUDU									
UMACNSVC	000002	DMKDIR	DMKLOG										
UMACOPT	000015	DMKDIR	DMKLOG	DMKUDU									
UMACOPT2	000002	DMKDIR	DMKLOG										
UMACPRIR	000004	DMKDIR	DMKLOG	DMKUDU									
UMACPUID	000003	DMKDIR	DMKLOG	DMKUDU									
UMACRT	000002	DMKDIR	DMKLOG										
UMACSIZE	000005	DMKDIR	DMKUDR										
UMACVROP	000002	DMKDIR	DMKLOG										
UMDISKAD	000001	DMKUDU											
UMDISKMD	000001	DMKUDU											
UMDISKMP	000001	DMKUDU											
UMDISKRP	000002	DMKUDU											
UMDISKWP	000001	DMKUDU											
UNEWPASS	000001	DMKUDU											
UNFIN	000005	DMKMIA											
UNOUPF	000002	DMKUDU											
UNSHRVM	000001	DMKCPU											
UOBJVMBK	000002	DMKUDU											
UOP	000005	DMKUDU											
UOPTIONS	000001	DMKUDU											
UPRIOR	000002	DMKUDU											
UPRIV	000003	DMKUDU											
URECMP	000003	DMKUDU											
URETCODE	000013	DMKUDU											
URPAGDEV	000003	DMKUDU											
URPAGDIR	000003	DMKUDU											
URPAGMAC	000002	DMKUDU											
USERCARD	000002	DMKACO	DMKCKP										
USERCL	000004	DMKMCC	DMKMNI	DMKMON									
USTORAGE	000002	DMKUDU											
USVDASD	000003	DMKUDU											
UTESTMD	000016	DMKUDU											
UUSERID	000003	DMKUDU											
UVMBLOK	000002	DMKUDU											
UVPAGBUF	000007	DMKUDU											
UVPAGDIR	000003	DMKUDU											
UWORK	000023	DMKUDU											
VCHADD	000045	DMKCFM	DMKCFP	DMKCPB	DMKCQG	DMKCSP	DMKCSU	DMKCSV	DMKDEF	DMKDIA	DMKDSP	DMKLOG	DMKSCN
		DMKSPL	DMKSSS	DMKVCH	DMKVCN	DMKVDA	DMKVDD	DMKVDS	DMKVIO	DMKVSI	DMKVSP		
VCHBLOK	000063	DMKCFM	DMKCFP	DMKCKP	DMKCPB	DMKCPV	DMKCQG	DMKCSP	DMKCSU	DMKCSV	DMKDEF	DMKDIA	DMKDSP
		DMKLNK	DMKLOG	DMKPRV	DMKSCN	DMKSPL	DMKSSS	DMKUSO	DMKVCH	DMKVCN	DMKVDA	DMKVDC	DMKVDD
		DMKVDS	DMKVIO	DMKVSI	DMKVSP								
VCHBMX	000011	DMKDEF	DMKPRV	DMKVCH	DMKVDS	DMKVIO	DMKVSI	DMKVSP					
VCHBUSY	000010	DMKCFP	DMKDSP	DMKVIO	DMKVSI								
VCHCEDEV	000004	DMKCFP	DMKDSP	DMKVIO	DMKVSP								

LABEL	COUNT	REFERENCES											
VCHCEPND	000010	DMKCFP	DMKDSP	DMKVIO	DMKVSI	DMKVSP							
VCHCUINT	000013	DMKCFM	DMKCFP	DMKCPB	DMKDSP	DMKSSS	DMKVCN	DMKVIO	DMKVSI	DMKVSP			
VCHCUTBL	000037	DMKCFM	DMKCFP	DMKCKP	DMKCPV	DMKCQG	DMKCSP	DMKCSU	DMKCSV	DMKDEF	DMKDIA	DMKDSP	DMKPRV
		DMKSCN	DMKSPL	DMKVCH	DMKVDC	DMKVDD	DMKVDS	DMKVIO	DMKVSI	DMKVSP			
VCHDED	000010	DMKCFP	DMKDEF	DMKLNK	DMKVCH	DMKVDA	DMKVDC	DMKVDD	DMKVSI				
VCHSEL	000017	DMKDEF	DMKDSP	DMKPRV	DMKVCH	DMKVDS	DMKVIO	DMKVSI	DMKVSP				
VCHSIZE	000011	DMKLOG	DMKUSO	DMKVCH	DMKVDC	DMKVDS							
VCHSTAT	000030	DMKCFP	DMKDEF	DMKDSP	DMKLNK	DMKVCH	DMKVDA	DMKVDC	DMKVDD	DMKVIO	DMKVSI	DMKVSP	
VCHTYPE	000024	DMKDEF	DMKDSP	DMKPRV	DMKVCH	DMKVDS	DMKVIO	DMKVSI	DMKVSP				
VCONADDR	000005	DMKVCN											
VCONBFSZ	000004	DMKVCN	DMKVDR										
VCONBUF	000010	DMKVCN	DMKVDR										
VCONCAW	000006	DMKVCN											
VCONCCW	000014	DMKVCN											
VCONCNT	000006	DMKVCN											
VCONCOMD	000020	DMKVCN											
VCONCTL	000006	DMKALG	DMKCFP	DMKGRF	DMKRGA	DMKVCN	DMKVDR						
VCONDWC	000007	DMKVCN											
VCONFLAG	000026	DMKVCN											
VCONIDAP	000003	DMKVCN											
VCONRBSZ	000006	DMKALG	DMKCFP	DMKGRF	DMKRGA	DMKVCN	DMKVDR						
VCONRBUF	000014	DMKALG	DMKCFP	DMKGRF	DMKRGA	DMKVCN	DMKVDR						
VCONRCNT	000005	DMKALG	DMKGRF	DMKRGA	DMKVCN								
VCONSIZE	000003	DMKVDR	DMKVDS										
VCONWBSZ	000005	DMKCFP	DMKVCN	DMKVDR									
VCONWBUF	000009	DMKCFP	DMKVCN	DMKVDR									
VCONWCNT	000002	DMKVCN											
VCUACTV	000012	DMKCFP	DMKVIO	DMKVSI									
VCUADD	000032	DMKCFM	DMKCFP	DMKCPB	DMKCQG	DMKCSP	DMKCSU	DMKCSV	DMKDEF	DMKDIA	DMKDSP	DMKLOG	DMKSCN
		DMKSPL	DMKSSS	DMKVCH	DMKVCN	DMKVDD	DMKVDS	DMKVIO	DMKVSI	DMKVSP			
VCUBLOK	000048	DMKCFM	DMKCFP	DMKCKP	DMKCPB	DMKCPV	DMKCQG	DMKCSP	DMKCSU	DMKCSV	DMKDEF	DMKDIA	DMKDSP
		DMKLOG	DMKNLD	DMKPRV	DMKSCN	DMKSPL	DMKSSS	DMKUSO	DMKVCH	DMKVCN	DMKVDA	DMKVDC	DMKVDD
		DMKVDS	DMKVIO	DMKVSI	DMKVSP								
VCUBUSY	000009	DMKCFP	DMKVIO	DMKVSI									
VCUCEPND	000004	DMKCFP	DMKDSP	DMKVCN	DMKVIO								
VCUCHBSY	000004	DMKCFP	DMKVIO										
VCUCTCA	000011	DMKDEF	DMKDSP	DMKVDS	DMKVIO	DMKVSI							
VCUCUEPN	000004	DMKCFP	DMKVIO	DMKVSI									
VCUDVINT	000013	DMKCFM	DMKCFP	DMKCPB	DMKDSP	DMKSSS	DMKVCN	DMKVIO	DMKVSI	DMKVSP			
VCUDVTBL	000046	DMKCFM	DMKCFP	DMKCKP	DMKCPV	DMKCQG	DMKCSP	DMKCSU	DMKCSV	DMKDEF	DMKDIA	DMKDSP	DMKNLD
		DMKPRV	DMKSCN	DMKSPL	DMKUSO	DMKVCH	DMKVDA	DMKVDC	DMKVDD	DMKVDS	DMKVIO	DMKVSI	DMKVSP
VCUINTS	000012	DMKCFP	DMKDSP	DMKVIO	DMKVSI								
VCUSHRD	000006	DMKDSP	DMKVCN	DMKVDS	DMKVIO	DMKVSI							
VCUSIZE	000013	DMKLOG	DMKUSO	DMKVCH	DMKVDC	DMKVDS							
VCUSTAT	000029	DMKCFP	DMKDSP	DMKVCN	DMKVIO	DMKVSI							
VCUTYPE	000014	DMKDEF	DMKDSP	DMKVCN	DMKVDS	DMKVIO	DMKVSI						

LABEL	COUNT	REFERENCES											
VDEVADD	000052	DMKCFM	DMKCFP	DMKCPB	DMKCQG	DMKCQP	DMKCSP	DMKCSQ	DMKCST	DMKCSU	DMKCSV	DMKDEF	DMKDIA
		DMKDSP	DMKLOG	DMKNLD	DMKSCN	DMKSPL	DMKSSS	DMKUSO	DMKVCH	DMKVCN	DMKVDC	DMKVDD	DMKVDS
		DMKVIO	DMKVSI	DMKVSP									
VDEVATTN	000008	DMKVCN											
VDEVAUCR	000002	DMKALG	DMKCFP										
VDEVBLOK	000128	DMKACO	DMKALG	DMKCCH	DMKCCW	DMKCFG	DMKCFH	DMKCFM	DMKCFP	DMKCKP	DMKCPB	DMKCPS	DMKCPV
		DMKCQG	DMKCQP	DMKCSB	DMKCSP	DMKCSQ	DMKCST	DMKCSU	DMKCSV	DMKDAS	DMKDEF	DMKDGD	DMKDIA
		DMKDIB	DMKDRD	DMKDSP	DMKGIO	DMKGRF	DMKHVC	DMKHVD	DMKIOS	DMKLNK	DMKLOG	DMKNLD	DMKPRV
		DMKQCN	DMKRGA	DMKSCN	DMKSPL	DMKSSS	DMKTHI	DMKTRC	DMKTRD	DMKTRK	DMKUNT	DMKUSO	DMKVCA
		DMKVCH	DMKVCN	DMKVDA	DMKVDC	DMKVDD	DMKVDR	DMKVDS	DMKVER	DMKVIO	DMKVSI	DMKVSP	DMKVSQ
VDEVBND	000019	DMKACO	DMKCCW	DMKCKP	DMKCQG	DMKCQP	DMKDGD	DMKLNK	DMKSSS	DMKVDR	DMKVDS	DMKVSI	
VDEVBUSY	000038	DMKCFM	DMKCFP	DMKCPB	DMKDGD	DMKDRD	DMKGIC	DMKSSS	DMKVCN	DMKVIO	DMKVSI	DMKVSP	
VDEVCATT	000004	DMKVDA	DMKVDD	DMKVDR									
VDEVCCW1	000028	DMKCFP	DMKVCA	DMKVCN	DMKVSP								
VDEVCFCL	000006	DMKVSP	DMKVSQ										
VDEVCFLG	000020	DMKALG	DMKCFP	DMKVCN									
VDEVCHAN	000017	DMKCFP	DMKDGD	DMKDSP	DMKGIO	DMKVCN	DMKVIC	DMKVSI	DMKVSP				
VDEVCHBS	000016	DMKCFM	DMKCFP	DMKVCN	DMKVIO	DMKVSI	DMKVSP						
VDEVCLAS	000022	DMKCKP	DMKCQG	DMKCSP	DMKCSU	DMKCSV	DMKDRD	DMKSPL	DMKVDS	DMKVSP	DMKVSQ		
VDEVCON	000007	DMKALG	DMKCFP	DMKGRF	DMKRGA	DMKVCN	DMKVDR	DMKVDS					
VDEVCONT	000012	DMKCQG	DMKCSP	DMKCSQ	DMKDRD	DMKVSP							
VDEVCOPY	000006	DMKCKP	DMKCQG	DMKCSP	DMKSPL	DMKVDS							
VDEVCPEX	000008	DMKCCW	DMKCFP	DMKDGD	DMKGIO								
VDEVCSPL	000015	DMKCFP	DMKCPS	DMKCQG	DMKCSP	DMKDAS	DMKQCN	DMKSPL	DMKVCN	DMKVDS	DMKVSP		
VDEVCSW	000113	DMKCFP	DMKCSP	DMKCSU	DMKCSV	DMKDSP	DMKGIC	DMKSPL	DMKTRC	DMKTRK	DMKUNT	DMKVCN	DMKVIO
		DMKVSI	DMKVSP										
VDEVCUE	000011	DMKCFP	DMKDSP	DMKVIO	DMKVSI								
VDEVDED	000062	DMKCCW	DMKCFP	DMKCKP	DMKCPB	DMKCPV	DMKCQG	DMKCSB	DMKCSP	DMKCSQ	DMKCST	DMKDEF	DMKDGD
		DMKDIA	DMKGIO	DMKHVD	DMKPRV	DMKSCN	DMKTRD	DMKVDD	DMKVDR	DMKVDS	DMKVER	DMKVIO	DMKVSI
		DMKVSP											
VDEVDET	000003	DMKSCN	DMKVDR										
VDEVDIAG	000008	DMKDRD	DMKVSP										
VDEVDIAL	000017	DMKCCW	DMKCFP	DMKDIA	DMKDIB	DMKNLD	DMKVSI						
VDEVDLY	000003	DMKSPL	DMKVSP										
VDEVENAB	000012	DMKCCW	DMKCFP	DMKCQG	DMKDIA	DMKDIB	DMKVSI						
VDEVEOF	000009	DMKCQG	DMKCSP	DMKVDS	DMKVSP								
VDEVEXTN	000013	DMKCKP	DMKCQG	DMKCSP	DMKCST	DMKSPL	DMKVDR	DMKVDS					
VDEVFCBK	000013	DMKCSB	DMKVDR	DMKVSP									
VDEVFEED	000007	DMKCFP	DMKVSP										
VDEVFLAG	000131	DMKACO	DMKCCW	DMKCFP	DMKCKP	DMKCPS	DMKCPV	DMKCQG	DMKCQP	DMKCSP	DMKDAS	DMKDEF	DMKDGD
		DMKDIA	DMKDIB	DMKDSP	DMKGIO	DMKLNK	DMKNLD	DMKQCN	DMKSCN	DMKSPL	DMKUNT	DMKVCH	DMKVCN
		DMKVDC	DMKVDR	DMKVDS	DMKVIO	DMKVSI	DMKVSP						
VDEVFLG2	000041	DMKCCW	DMKCFP	DMKDEF	DMKDGD	DMKGIO	DMKLOG	DMKUNT	DMKVDA	DMKVDR	DMKVDS	DMKVSI	
VDEVFOR	000020	DMKCQG	DMKCSP	DMKSPL									
VDEVHOLD	000009	DMKCQG	DMKCSP	DMKSPL	DMKVSP								
VDEVINTS	000034	DMKCFM	DMKCFP	DMKCPB	DMKDSP	DMKSSS	DMKVCA	DMKVCN	DMKVIO	DMKVSI	DMKVSP		

LABEL	COUNT	REFERENCES											
VDEVIOB	000017	DMKCFP	DMKDGD	DMKDIA	DMKDIB	DMKGIO	DMKHVC	DMKVIO	DMKVSI				
VDEVIOCT	000008	DMKIOS	DMKVCA	DMKVCN	DMKVSP								
VDEVIOER	000027	DMKCCH	DMKCCW	DMKCFP	DMKDGD	DMKDSP	DMKGIO	DMKVIO	DMKVSI				
VDEVKEY	000013	DMKVCN	DMKVSP										
VDEVLINK	000025	DMKCFP	DMKDEF	DMKSCN	DMKUNT	DMKVCH	DMKVDC	DMKVDR	DMKVDS				
VDEVNRDY	000027	DMKCFP	DMKCPB	DMKCQG	DMKDIA	DMKVCA	DMKVCN	DMKVDS	DMKVIO	DMKVSI	DMKVSP		
VDEVODE	000007	DMKCFP	DMKUNT	DMKVSI									
VDEVPEND	000020	DMKCFM	DMKCFP	DMKCPB	DMKCSP	DMKCSU	DMKCSV	DMKDGD	DMKDSP	DMKGIO	DMKSPL	DMKSSS	DMKVCN
		DMKVIO	DMKVSI	DMKVSP									
VDEVPOSN	000016	DMKCCW	DMKDEF	DMKDGD	DMKGIO	DMKVIO							
VDEVPOST	000005	DMKDSP	DMKVIO	DMKVSI									
VDEVPURG	000010	DMKCSP	DMKCSQ	DMKVSP	DMKVSQ								
VDEVRDO	000014	DMKCCW	DMKCQG	DMKCQP	DMKDGD	DMKLNK	DMKSCN	DMKVDS	DMKVSI				
VDEVREAL	000069	DMKACO	DMKCCW	DMKCFG	DMKCFH	DMKCFP	DMKCKP	DMKCPB	DMKCQG	DMKCQP	DMKDGD	DMKDIA	DMKHVD
		DMKIOS	DMKLNK	DMKLOG	DMKPRV	DMKSCN	DMKTHI	DMKTRD	DMKTRK	DMKUNT	DMKVCA	DMKVDD	DMKVDR
		DMKVDS	DMKVER	DMKVSI									
VDEVRELN	000034	DMKCCW	DMKCFG	DMKCFH	DMKCQP	DMKDEF	DMKDGD	DMKGIO	DMKLNK	DMKSCN	DMKSSS	DMKUNT	DMKVDR
		DMKVDS	DMKVER	DMKVIO									
VDEVRES	000009	DMKCCW	DMKCFP	DMKDGD	DMKGIO	DMKUNT	DMKVSI						
VDEVRRB	000011	DMKCCW	DMKCFP	DMKDGD	DMKGIO	DMKUNT	DMKVDR	DMKVDS	DMKVSI				
VDEVRRF	000011	DMKCCW	DMKCFP	DMKDGD	DMKGIO	DMKVDR	DMKVDS	DMKVSI					
VDEVRSRL	000003	DMKCCW	DMKCFP	DMKVDS									
VDEVSAS	000004	DMKCCW	DMKVIO										
VDEVSFLG	000070	DMKCFP	DMKCKP	DMKCQG	DMKCSP	DMKCSQ	DMKDRD	DMKQCN	DMKSPL	DMKVCN	DMKVDD	DMKVDS	DMKVSP
		DMKVSQ											
VDEVSIZE	000025	DMKCQP	DMKCSP	DMKCSQ	DMKCST	DMKLOG	DMKSCN	DMKUSO	DMKVCH	DMKVDC	DMKVDS		
VDEVSNSE	000029	DMKVCN	DMKVSP										
VDEVSPL	000036	DMKCFP	DMKCKP	DMKCPS	DMKCSP	DMKCSQ	DMKCSU	DMKCSV	DMKDRD	DMKSPL	DMKVDR	DMKVSI	DMKVSP
		DMKVSQ											
VDEVSTAT	000176	DMKCCW	DMKCFM	DMKCFP	DMKCKP	DMKCPB	DMKCPV	DMKCQG	DMKCSB	DMKCSP	DMKCSQ	DMKCST	DMKCSU
		DMKCSV	DMKDEF	DMKDGD	DMKDIA	DMKDRD	DMKDSB	DMKGIO	DMKHVD	DMKPRV	DMKSCN	DMKSPL	DMKSSS
		DMKTRD	DMKVCA	DMKVCN	DMKVDA	DMKVDD	DMKVDR	DMKVDS	DMKVER	DMKVIO	DMKVSI	DMKVSP	
VDEVSVC	000019	DMKVDD	DMKVSP										
VDEVTDSK	000014	DMKACO	DMKCKP	DMKCPV	DMKCQG	DMKCQP	DMKDEF	DMKVCH	DMKVDC	DMKVDR	DMKVDS		
VDEVTERM	000011	DMKCQG	DMKCSP	DMKQCN	DMKVCN	DMKVDS							
VDEVTIC	000006	DMKVCN											
VDEVTMAT	000004	DMKACO	DMKCKP	DMKVDS									
VDEVTRAN	000003	DMKVCN											
VDEVTYPC	000169	DMKACO	DMKCCW	DMKCFP	DMKCKP	DMKCPB	DMKCPV	DMKCQG	DMKCQP	DMKCSP	DMKCSQ	DMKCST	DMKCSU
		DMKCSV	DMKDEF	DMKDGD	DMKDIA	DMKDRD	DMKGIC	DMKHVD	DMKLNK	DMKPRV	DMKSCN	DMKSPL	DMKTRD
		DMKVCH	DMKVCN	DMKVDC	DMKVDD	DMKVDR	DMKVDS	DMKVER	DMKVIO	DMKVSI	DMKVSP		
VDEVTYPE	000146	DMKCCW	DMKCFP	DMKCKP	DMKCPB	DMKCQG	DMKCSB	DMKCSP	DMKCSQ	DMKCST	DMKCSU	DMKCSV	DMKDGD
		DMKDIA	DMKDRD	DMKHVD	DMKLNK	DMKSCN	DMKSPL	DMKUNT	DMKVCN	DMKVDR	DMKVDS	DMKVER	DMKVIO
		DMKVSI	DMKVSP	DMKVSQ									
VDEVUC	000011	DMKCCW	DMKCFP	DMKDGD	DMKGIO	DMKVIO	DMKVSI						
VDEVUNIT	000004	DMKVSP											

LABEL	COUNT	REFERENCES											
VDEVUSER	000006	DMKCFP	DMKLNK	DMKSCN	DMKUNT	DMKVDS							
VDEVVCF	000003	DMKVCN											
VDEVXFER	000021	DMKCKP	DMKCQG	DMKCSP	DMKSPL								
VDEV231B	000008	DMKCCW	DMKCQG	DMKUNT									
VDEV231T	000003	DMKCCW	DMKCQG										
VFAULT	000003	DMKPRG	DMKPTR										
VFCBBLOK	000012	DMKCSB	DMKVSP										
VFCBCHL	000005	DMKVSP											
VFCBCNT	000009	DMKCSB	DMKVSP										
VFCBEOF	000003	DMKVSP											
VFCBFLAG	000009	DMKVSP											
VFCBLOAD	000009	DMKCSB	DMKVSP										
VFCBNDEX	000008	DMKCSB	DMKVSP										
VFCBSIZE	000006	DMKCSB	DMKVDR	DMKVSP									
VIRTUAL	000041	DMKCCW	DMKCFP	DMKCQP	DMKDEF	DMKDGD	DMKDIR	DMKDSB	DMKIOS	DMKLNK	DMKLOG	DMKSSS	DMKVDA
		DMKVSI											
VMABLOK	000006	DMKATS	DMKCFG	DMKPGS	DMKVMA								
VMACCOUN	000003	DMKHVD	DMKLOG	DMKUSO									
VMACNT	000007	DMKACO	DMKCKP	DMKJRL	DMKLOG	DMKUDU	DMKUSO						
VMACOUNT	000010	DMKACO	DMKCKP	DMKHVD	DMKLOG	DMKSPL	DMKUSO						
VMACTDEV	000005	DMKDGD	DMKGIO	DMKTHI	DMKVSI								
VMADSTOP	000009	DMKCFD	DMKCFS	DMKPGS	DMKSVC								
VMAEX	000010	DMKBLD	DMKCFO	DMKMON	DMKSCH								
VMAEXP	000009	DMKCFO	DMKSCH	DMKUSO									
VMAFF	000018	DMKCFS	DMKCPU	DMKCQR	DMKDSP	DMKLOG	DMKMCT						
VMAFFON	000009	DMKCFS	DMKCPU	DMKCQR	DMKDSP	DMKMCT							
VMAFPNT	000010	DMKATS	DMKCFG	DMKPGS	DMKVMA								
VMANAME	000003	DMKATS	DMKCFG	DMKPGS									
VMAPTIME	000010	DMKACO	DMKAPI	DMKBLD	DMKCPI	DMKLOG	DMKTMR						
VMASHRBK	000003	DMKCFG	DMKVMA										
VMASIZE	000004	DMKATS	DMKCFG	DMKPGS									
VMASSIST	000007	DMKATS	DMKCFG	DMKPGS	DMKVMA								
VMBADCR0	000004	DMKVAT											
VMBCAUTH	000012	DMKCPP	DMKVMC										
VMBLOK	000884	DMKACO	DMKALG	DMKAPI	DMKATS	DMKBLD	DMKCCH	DMKCCW	DMKCDB	DMKCDM	DMKCDS	DMKCFC	DMKCFD
		DMKCFG	DMKCFH	DMKCFM	DMKCFO	DMKCFP	DMKCFS	DMKCFT	DMKCKP	DMKCKS	DMKCNS	DMKCPB	DMKCPI
		DMKCPS	DMKCPU	DMKCPV	DMKCQG	DMKCQH	DMKCQP	DMKCQR	DMKCQY	DMKCSB	DMKCSO	DMKCSP	DMKCSQ
		DMKCST	DMKCSU	DMKCSV	DMKDAS	DMKDEF	DMKDGD	DMKDIA	DMKDIB	DMKDRD	DMKDSP	DMKENT	DMKERM
		DMKEXT	DMKFRE	DMKGIO	DMKGRF	DMKGRT	DMKHVC	DMKHVD	DMKICE	DMKIOF	DMKIOG	DMKIOS	DMKISM
		DMKJRL	DMKLNK	DMKLOG	DMKLOH	DMKLOK	DMKMCC	DMKMCD	DMKMCH	DMKMCT	DMKMIA	DMKMID	DMKMNI
		DMKMON	DMKMSG	DMKMSW	DMKNES	DMKNET	DMKNLD	DMKNLE	DMKPAG	DMKPER	DMKPGS	DMKPGT	DMKPRG
		DMKPRV	DMKPSA	DMKPTR	DMKQCN	DMKRGA	DMKRGB	DMKRNH	DMKRPA	DMKRSE	DMKRSP	DMKSCH	DMKSCN
		DMKSEP	DMKSNC	DMKSPL	DMKSSS	DMKSTK	DMKSVC	DMKTCS	DMKTHI	DMKTMR	DMKTRA	DMKTRC	DMKTRD
		DMKTRK	DMKUDR	DMKUDU	DMKUNT	DMKUSO	DMKVAT	DMKVCA	DMKVCH	DMKVCN	DMKVDA	DMKVDC	DMKVDD
		DMKVDR	DMKVDS	DMKVER	DMKVIO	DMKVMA	DMKVMC	DMKVSI	DMKVSP	DMKVSQ	DMKWRM		
VMBSIZE	000006	DMKBLD	DMKDIA	DMKLOG	DMKUSO								

LABEL	COUNT	REFERENCES											
VMCAAUTS	000002	DMKVMC											
VMCACNT	000007	DMKVMC											
VMCAPRTY	000002	DMKVMC											
VMCAQIES	000004	DMKVMC											
VMCASTAT	000008	DMKVMC											
VMCBLOK	000044	DMKDSP	DMKVMC										
VMCBSIZE	000008	DMKVMC											
VMCCBUSY	000008	DMKVMC											
VMCCRECP	000003	DMKVMC											
VMCCSTAT	000019	DMKDSP	DMKVMC										
VMCCXINT	000008	DMKDSP	DMKVMC										
VMCEFLG	000026	DMKVMC											
VMCF	000029	DMKALG	DMKCFM	DMKCNS	DMKDIA	DMKDSP	DMKEMC	DMKGRF	DMKHVC	DMKLNK	DMKLOG	DMKQCN	DMKRGA
		DMKRSE	DMKVCN										
VMCFPNT	000016	DMKDSP	DMKVMC										
VMCFREAD	000006	DMKCFM	DMKDSP	DMKLOG	DMKQCN								
VMCFRUN	000008	DMKCFM	DMKCFS	DMKCQR	DMKDSP	DMKPRG							
VMCFUNC	000005	DMKVMC											
VMCFWAIT	000035	DMKACO	DMKALG	DMKBLD	DMKCFM	DMKCNS	DMKDGD	DMKDSP	DMKGRF	DMKHVC	DMKLOG	DMKPRG	DMKQCN
		DMKRGA	DMKRNH	DMKTRA	DMKTRC	DMKTRD							
VMCHCNT	000012	DMKLOG	DMKUSO	DMKVCH	DMKVDC	DMKVDS							
VMCHSTRT	000062	DMKCFM	DMKCFP	DMKCKP	DMKCPV	DMKCQG	DMKCSP	DMKCSU	DMKCSV	DMKDEF	DMKDIA	DMKDSP	DMKLOG
		DMKSCN	DMKSPL	DMKUSO	DMKVCH	DMKVCN	DMKVDC	DMKVDS	DMKVIO	DMKVSI	DMKVSP		
VMCHTBL	000029	DMKBLD	DMKCFM	DMKCFP	DMKCKP	DMKCPV	DMKCQG	DMKCSP	DMKCSU	DMKCSV	DMKDEF	DMKDIA	DMKDSP
		DMKSCN	DMKSPL	DMKUSO	DMKVCH	DMKVDD	DMKVDS	DMKVSP					
VMCKEY	000006	DMKVMC											
VMCLASSA	000012	DMKCFC	DMKCFM	DMKCFO	DMKCFS	DMKDEF	DMKHVD	DMKMSG	DMKNET	DMKTHI			
VMCLASSB	000015	DMKCFC	DMKCFM	DMKCFO	DMKCFS	DMKDEF	DMKHVD	DMKMSG	DMKNET	DMKTHI	DMKVDC		
VMCLASSC	000014	DMKCFC	DMKCFM	DMKCFO	DMKCFS	DMKHVD	DMKNET	DMKTHI					
VMCLASSD	000018	DMKCFC	DMKCFM	DMKCFO	DMKCFS	DMKCQH	DMKCQR	DMKCSU	DMKCSV	DMKNET	DMKTHI		
VMCLASSE	000011	DMKCFC	DMKCFM	DMKCFO	DMKCFS	DMKHVD	DMKNET	DMKTHI					
VMCLASSF	000013	DMKCCW	DMKCFC	DMKCFM	DMKCFO	DMKCFS	DMKHVD	DMKIOE	DMKNET	DMKTHI			
VMCLASSG	000007	DMKCFC	DMKCFM	DMKCFO	DMKCFS	DMKCQG	DMKNET	DMKTHI					
VMCLASSH	000002	DMKCFC	DMKCFM										
VMCLENA	000008	DMKVMC											
VMCLEVEL	000040	DMKCCW	DMKCFC	DMKCFS	DMKCQG	DMKCQH	DMKCQR	DMKCSU	DMKCSV	DMKDEF	DMKHVD	DMKIOE	DMKLOG
		DMKMSG	DMKNET	DMKTHI	DMKVDC								
VMCMEFLG	000001	DMKMSG											
VMCMFUNC	000001	DMKMSG											
VMCMHDR	000001	DMKMSG											
VMCMID	000001	DMKVMC											
VMCMLEN	000009	DMKMSG	DMKVMC										
VMCMLENA	000001	DMKMSG											
VMCMMID	000001	DMKMSG											
VMCMUSE	000003	DMKMSG											
VMCMUSER	000001	DMKMSG											

LABEL	COUNT	REFERENCES											
VMCMVADA	000001	DMKMSG											
VMCOMND	000010	DMKALG	DMKCFC	DMKCFG	DMKCSU	DMKCSV	DMKHVC	DMKLNK	DMKQCN	DMKUSO			
VMCOMP	000005	DMKDGD	DMKDSP	DMKGIO	DMKSCH								
VMCONBUF	000006	DMKBLD	DMKHVC	DMKQCN									
VMCONLN	000004	DMKHVC	DMKQCN										
VMCPARM	000003	DMKVMC											
VMCPAUTS	000001	DMKVMC											
VMCPFLG1	000005	DMKVMC											
VMCPFUNC	000001	DMKVMC											
VMCPIDEN	000002	DMKVMC											
VMCPLEN	000003	DMKHVC	DMKVMC										
VMCPMID	000001	DMKVMC											
VMCPNT	000019	DMKDSP	DMKVMC										
VMCPPRTY	000003	DMKVMC											
VMCPRTY	000003	DMKVMC											
VMCPSENR	000002	DMKVMC											
VMCPSENX	000002	DMKMSG	DMKVMC										
VMCPSMSG	000001	DMKVMC											
VMCPTIME	000010	DMKACO	DMKAPI	DMKBLD	DMKCPI	DMKDSP	DMKLOG	DMKTMR					
VMCPUID	000006	DMKCFS	DMKCQY	DMKLOG	DMKPRV								
VMCPUSE	000003	DMKVMC											
VMCPUSER	000007	DMKVMC											
VMCPUTMR	000019	DMKCFP	DMKPSA	DMKSCH	DMKTMR								
VMCPVADA	000007	DMKVMC											
VMCPVADB	000001	DMKVMC											
VMCPWAIT	000007	DMKCFM	DMKDSP	DMKPTR	DMKSCH								
VMCRDS	000006	DMKACO	DMKMON	DMKTHI	DMKVSP								
VMCRESP	000013	DMKVMC											
VMCRJCT	000001	DMKVMC											
VMCSMAX	000001	DMKVMC											
VMCSTAT	000018	DMKVMC											
VMCTOD	000006	DMKVMC											
VMCUCNT	000012	DMKLOG	DMKUSO	DMKVCH	DMKVDC	DMKVDS							
VMCUSE	000003	DMKVMC											
VMCUSER	000020	DMKVMC											
VMCUSTRT	000062	DMKCFM	DMKCFP	DMKCKP	DMKCPV	DMKCQG	DMKCSE	DMKCSU	DMKCSV	DMKDEF	DMKDIA	DMKDSP	DMKLOG
		DMKPRV	DMKSCN	DMKSPL	DMKVCH	DMKVCN	DMKVDC	DMKVDS	DMKVIO	DMKVSI	DMKVSP		
VMCVADA	000009	DMKVMC											
VMCVADB	000002	DMKVMC											
VMCXCODE	000004	DMKDSP	DMKVMC										
VMCXMASK	000002	DMKVMC											
VMCXSTAT	000012	DMKCFP	DMKVMC										
VMC01	000003	DMKVMC											
VMC02	000001	DMKVMC											
VMC03	000001	DMKVMC											
VMC04	000001	DMKVMC											

LABEL	COUNT	REFERENCES											
VMC05	000004	DMKVMC											
VMC06	000003	DMKVMC											
VMC07	000002	DMKVMC											
VMC08	000001	DMKVMC											
VMC09	000001	DMKVMC											
VMC10	000001	DMKVMC											
VMC11	000002	DMKVMC											
VMC12	000001	DMKVMC											
VMC13	000001	DMKVMC											
VMC14	000001	DMKVMC											
VMC15	000003	DMKVMC											
VMC16	000003	DMKVMC											
VMC17	000003	DMKVMC											
VMC18	000001	DMKVMC											
VMC19	000008	DMKVMC											
VMC20	000001	DMKVMC											
VMDEFSTK	000012	DMKCPU	DMKDSP	DMKLOK	DMKSTK	DMKTHI							
VMDELAY	000013	DMKCFC	DMKCFM	DMKDIA	DMKLOG	DMKQCN	DMKUSO						
VMDFTPNT	000021	DMKAPI	DMKBLD	DMKCPI	DMKCPU	DMKDIA	DMKDSP	DMKLOG	DMKPRG	DMKPRV	DMKSVC	DMKTMR	DMKUSO
		DMKVAT	DMKVSI										
VMDISC	000030	DMKALG	DMKCQG	DMKCQP	DMKCQY	DMKJRL	DMKLOG	DMKMSG	DMKMSW	DMKPSA	DMKQCN	DMKUSO	DMKVCN
VMDIST	000005	DMKCKP	DMKCQG	DMKLOG	DMKSPL	DMKVDS							
VMDROP1	000003	DMKSCH											
VMDSP	000027	DMKDSP	DMKPGS	DMKPRG	DMKPRV	DMKPSA	DMKPTR	DMKSCH	DMKTMR				
VMDSTAT	000089	DMKACO	DMKCFP	DMKCPV	DMKDSP	DMKIOS	DMKPAG	DMKPGS	DMKPRG	DMKPRV	DMKPSA	DMKPTR	DMKSCH
		DMKTHI	DMKTMR	DMKVCN	DMKVIO	DMKVSI							
VMDVCNT	000017	DMKCQP	DMKCSP	DMKCSQ	DMKCST	DMKLOG	DMKSCN	DMKUSO	DMKVCH	DMKVDC	DMKVDS		
VMDVSTRT	000238	DMKALG	DMKCCH	DMKCCW	DMKCFM	DMKCFP	DMKCFT	DMKCKP	DMKCPS	DMKCPV	DMKCQG	DMKCQP	DMKCSP
		DMKCSQ	DMKCST	DMKCSU	DMKCSV	DMKDAS	DMKDEF	DMKDGD	DMKDIA	DMKDIB	DMKDRD	DMKDSP	DMKGIO
		DMKGRF	DMKHVC	DMKHVD	DMKLOG	DMKPRV	DMKQCN	DMKRGA	DMKSCN	DMKSPL	DMKUSO	DMKVCA	DMKVCH
		DMKVCN	DMKVDC	DMKVDR	DMKVDS	DMKVIO	DMKVSI	DMKVSP	DMKVSQ				
VMECEXT	000050	DMKBLD	DMKCDB	DMKCDM	DMKCDS	DMKCFG	DMKCFH	DMKCFP	DMKCFS	DMKDSP	DMKEXT	DMKLOG	DMKPRG
		DMKPRV	DMKSCH	DMKSVC	DMKTMR	DMKTRC	DMKTRD	DMKUSO	DMKVAT	DMKVSI			
VMELIG	000009	DMKPTR	DMKSCH	DMKTHI									
VMEPRIOR	000006	DMKBLD	DMKMON	DMKSCH									
VMESTAT	000129	DMKBLD	DMKCDB	DMKCDM	DMKCDS	DMKCFD	DMKCFG	DMKCFP	DMKCFS	DMKCPB	DMKCPU	DMKDGD	DMKDRD
		DMKDSP	DMKGIO	DMKHVC	DMKHVD	DMKIOS	DMKLOG	DMKMCH	DMKPGS	DMKPRG	DMKPRV	DMKPSA	DMKPTR
		DMKRPA	DMKSVC	DMKTMR	DMKTRC	DMKTRD	DMKUDR	DMKVAT	DMKVCN	DMKVIO	DMKVSI	DMKVSP	
VMEXTCM	000066	DMKCDB	DMKCDM	DMKCDS	DMKCFG	DMKCPB	DMKDGD	DMKDRD	DMKDSP	DMKGIO	DMKHVC	DMKHVD	DMKIOS
		DMKMCH	DMKPRG	DMKPRV	DMKPSA	DMKSVC	DMKTMR	DMKTRC	DMKTRD	DMKUDR	DMKVCN	DMKVIO	DMKVSI
		DMKVSP											
VMEXTPND	000006	DMKDSP	DMKPRV	DMKVMC									
VMEXWAIT	000050	DMKACO	DMKCFP	DMKDGD	DMKDSP	DMKGIO	DMKHVC	DMKMCH	DMKMCT	DMKPRG	DMKPRV	DMKRPA	DMKSVC
		DMKTHI	DMKTMR	DMKTRA	DMKTRC	DMKTRD	DMKVAT	DMKVCN	DMKVER	DMKVIO	DMKVMC	DMKVSI	DMKVSP
VMFAUTO	000005	DMKCCW	DMKCFP	DMKCFS	DMKCQR								
VMFBMX	000007	DMKCQG	DMKDEF	DMKLOG	DMKVCH	DMKVDS							

LABEL	COUNT	REFERENCES											
VMFLPAG	000009	DMKPTR											
VMFPRS	000039	DMKCDB	DMKCDM	DMKCDS	DMKCFG	DMKDSP	DMKIOS	DMKMCH	DMKPRG	DMKPSA	DMKSVC		
VMFSTAT	000021	DMKCCW	DMKCFP	DMKCFS	DMKCQG	DMKCQR	DMKDEF	DMKHVC	DMKLNK	DMKLOG	DMKTRA	DMKVCH	DMKVDS
VMFVTMR	000005	DMKCFS	DMKLOG	DMKTRA									
VMGENIO	000023	DMKGRF	DMKQCN	DMKRGA	DMKRGB	DMKVCN							
VMGPRS	000068	DMKCDB	DMKCDM	DMKCDS	DMKCFG	DMKDGD	DMKDSP	DMKEXT	DMKGIO	DMKHVC	DMKHVD	DMKIOS	DMKMCH
		DMKPRG	DMKPRV	DMKPSA	DMKSSS	DMKSVC	DMKTMR	DMKTRC	DMKVER	DMKVSI			
VMGRFTAB	000011	DMKBLD	DMKCFT	DMKCQR	DMKGRF	DMKGRT							
VMHIPRI	000010	DMKCFO	DMKSCH										
VMIDLE	000022	DMKACO	DMKCCW	DMKCFO	DMKCFP	DMKDGD	DMKDSP	DMKGIO	DMKIOS	DMKSCH	DMKVCN	DMKVIO	DMKVSI
VMINHMIG	000003	DMKDEF	DMKUSO										
VMINQ	000031	DMKACO	DMKCFP	DMKCPV	DMKDSP	DMKIOS	DMKPAG	DMKPGS	DMKPRV	DMKPTR	DMKSCH	DMKTHI	DMKTMR
VMINST	000083	DMKHVC	DMKHVD	DMKMON	DMKPRV	DMKRPA	DMKSVC	DMKTMR	DMKTRD	DMKVSI	DMKVSP		
VMINVPAG	000025	DMKBLD	DMKCDB	DMKCDM	DMKCDS	DMKCFP	DMKCPU	DMKDSP	DMKMCH	DMKPGS	DMKPRV	DMKPTR	DMKRPA
		DMKTMR	DMKTRC	DMKTRD	DMKVAT								
VMINVSEG	000012	DMKCDB	DMKCDM	DMKCDS	DMKDSP	DMKPRV	DMKVAT						
VMIOACTV	000014	DMKCFP	DMKDSP	DMKVIO	DMKVSI								
VMIOCNT	000012	DMKACO	DMKCKP	DMKDGD	DMKGIO	DMKMON	DMKTHI	DMKVSI					
VMIOINT	000017	DMKCFM	DMKCFP	DMKCPB	DMKDSP	DMKPRV	DMKSCH	DMKSSS	DMKVCN	DMKVIO	DMKVSI	DMKVSP	
VMIOLOG	000008	DMKCCH	DMKCFP	DMKPRV									
VMIOPND	000017	DMKCFM	DMKCFP	DMKCPB	DMKDSP	DMKPRG	DMKPRV	DMKSSS	DMKVAT	DMKVCN	DMKVIO	DMKVSI	DMKVSP
VMIOWAIT	000034	DMKCFG	DMKCFP	DMKDGD	DMKDIB	DMKDSP	DMKGIC	DMKHVC	DMKICS	DMKPRG	DMKTHI	DMKVCA	DMKVIO
		DMKVSI											
VMISAM	000005	DMKCCW	DMKCFS	DMKCQR	DMKLOG								
VMKILL	000027	DMKALG	DMKCDB	DMKCDM	DMKCFM	DMKCFP	DMKDIA	DMKDSP	DMKJRL	DMKLNK	DMKLOG	DMKMCH	DMKMCT
		DMKMSG	DMKQCN	DMKUSO	DMKVDD								
VMLINS	000006	DMKACO	DMKMON	DMKTHI	DMKVSP								
VMLOCK	000042	DMKBLD	DMKCPU	DMKDSP	DMKFRE	DMKIOS	DMKLOK	DMKMCH	DMKPAG	DMKPTR			
VMLOCKER	000006	DMKBLD	DMKIOS	DMKLOK	DMKPTR								
VMLOGOFF	000046	DMKACO	DMKCDB	DMKCDM	DMKCFG	DMKCFH	DMKCFM	DMKCFP	DMKCNS	DMKCPV	DMKDGD	DMKDIA	DMKDSP
		DMKGRF	DMKMCT	DMKMSG	DMKPGS	DMKQCN	DMKRGA	DMKRGB	DMKSCN	DMKTRC	DMKTRD	DMKUSO	DMKVCA
		DMKVCN	DMKVDD	DMKVMA									
VMLOGON	000033	DMKBLD	DMKCFC	DMKCFM	DMKCKP	DMKCNS	DMKCPI	DMKCPV	DMKGRF	DMKLNK	DMKLOG	DMKMCT	DMKMON
		DMKQCN	DMKRGA	DMKRNH	DMKSCN	DMKUSO	DMKVDD						
VMLONGWT	000002	DMKACO	DMKSCH										
VMLOPRI	000003	DMKDGD	DMKGIO	DMKSCH									
VMLSTPRC	000004	DMKDSP	DMKMON										
VMMACCON	000005	DMKCFS	DMKCPV	DMKCQR	DMKLOG								
VMMADDR	000014	DMKCFS	DMKDSP	DMKMCH	DMKPRV	DMKPTR	DMKRPA	DMKTRA					
VMMCODE	000013	DMKBLD	DMKCFS	DMKCQR	DMKHVC	DMKLOG	DMKQCN	DMKVMI					
VMMCPAST	000003	DMKCFO	DMKLOG										
VMMCPENV	000014	DMKCFT	DMKCNS	DMKCQR	DMKGRF	DMKLOG	DMKRGA	DMKRNH					
VMMCR6	000039	DMKCFD	DMKCFO	DMKCFS	DMKCQR	DMKDSP	DMKLOG	DMKMCH	DMKPRV	DMKPTR	DMKRPA	DMKSVC	DMKTRA
VMMFE	000011	DMKAPI	DMKCFS	DMKCPI	DMKCQR	DMKDSP	DMKLOG						
VMMICRO	000009	DMKBLD	DMKCFS	DMKDSP	DMKLOG	DMKPRV	DMKUSO						
VMMICSVC	000007	DMKCFD	DMKCFP	DMKCFS	DMKLOG	DMKSVC							

LABEL	COUNT	REFERENCES											
VMMIMSG	000014	DMKCFS	DMKCQR	DMKCSU	DMKCSV	DMKDEF	DMKLOG	DMKVDD					
VMMLEVEL	000062	DMKBLD	DMKCFG	DMKCFM	DMKCFS	DMKCFT	DMKCNS	DMKCPV	DMKCQR	DMKCSV	DMKGRF	DMKHVC	DMKLOG
		DMKMID	DMKMSG	DMKQCN	DMKRGA	DMKRNH	DMKSPL	DMKUSO	DMKVCN				
VMMLINED	000010	DMKBLD	DMKCFS	DMKCQR	DMKGRF	DMKLOG	DMKMSG	DMKRGA	DMKVCN				
VMMLVL2	000014	DMKCFS	DMKCQR	DMKCSU	DMKCSV	DMKDEF	DMKLOG	DMKVDD					
VMMNOSK	000002	DMKDSP											
VMMPROB	000003	DMKDSP	DMKPRV										
VMMSGON	000015	DMKBLD	DMKCFS	DMKCQR	DMKCSV	DMKLOG	DMKMID	DMKMSG	DMKSPL	DMKUSO			
VMMSHADT	000002	DMKDSP											
VMMSTMP	000005	DMKCFM	DMKCFT	DMKQCN	DMKVCN								
VMMSVC	000009	DMKAPI	DMKCFD	DMKCFS	DMKCPI	DMKCQR	DMKLOG	DMKSVC					
VMMTEXT	000014	DMKBLD	DMKCFS	DMKCQR	DMKHVC	DMKLOG	DMKMSG	DMKQCN	DMKVMI				
VMMVTMR	000016	DMKCFS	DMKCQR	DMKDSP	DMKLOG	DMKMCH	DMKPTR	DMKRPA	DMKTRA				
VMM360	000003	DMKCFS	DMKLOG										
VMNDCNT	000007	DMKDSP	DMKPTR	DMKUSO									
VMNEWCRO	000011	DMKCDB	DMKCDM	DMKCDS	DMKDSP	DMKPRV	DMKVAT						
VMNORUN	000006	DMKACO	DMKDSP	DMKRPA	DMKSCH								
VMNOTRAN	000005	DMKCFP	DMKCFS	DMKCQR	DMKVSI								
VMNPWOCL	000004	DMKHVC	DMKLNK	DMKLOG									
VMNSHR	000003	DMKCFG	DMKPGS										
VMOSTAT	000157	DMKALG	DMKATS	DMKCCW	DMKCDB	DMKCDM	DMKCFC	DMKCFG	DMKCFM	DMKCFP	DMKCFS	DMKCNS	DMKCPU
		DMKCQG	DMKCQP	DMKCQR	DMKCQY	DMKCSO	DMKDIA	DMKDSP	DMKGRF	DMKHVC	DMKJRL	DMKLNK	DMKLOG
		DMKLOH	DMKLOK	DMKMCH	DMKMCT	DMKMSG	DMKMSW	DMKNES	DMKNET	DMKPGS	DMKPRG	DMKPSA	DMKPTR
		DMKQCN	DMKRGA	DMKRNH	DMKRSE	DMKUSO	DMKVAT	DMKVCN	DMKVDA	DMKVDD	DMKVDS	DMKVSQ	
VMPAGES	000051	DMKATS	DMKBLD	DMKCPI	DMKCPU	DMKCPV	DMKDSP	DMKMON	DMKPGS	DMKPTR	DMKRPA	DMKSCH	DMKTHI
		DMKUDR	DMKVMA										
VMPAGEX	000006	DMKCFP	DMKCFS	DMKCQR	DMKPRG	DMKVAT							
VMPA2APL	000006	DMKCFG	DMKCPB	DMKGRF	DMKHVD	DMKRGA							
VMPDISK	000010	DMKATS	DMKCPU	DMKMON	DMKPGS	DMKPGT	DMKTHI						
VMPDRUM	000011	DMKATS	DMKCPU	DMKMON	DMKPGS	DMKPGT	DMKTHI						
VMPEND	000060	DMKCFH	DMKCFM	DMKCFP	DMKCPB	DMKCPU	DMKDSP	DMKLOK	DMKPER	DMKPRG	DMKPRV	DMKSCH	DMKSSS
		DMKSTK	DMKSVC	DMKTHI	DMKTMR	DMKTRC	DMKVAT	DMKVCN	DMKVIO	DMKVSI	DMKVSP		
VMPERCM	000014	DMKDSP	DMKPRG	DMKPRV	DMKTMR								
VMPERPND	000024	DMKDSP	DMKPER	DMKPRG	DMKPRV	DMKSVC	DMKTMR	DMKTRC	DMKVAT				
VMPFUNC	000010	DMKCFS	DMKCQY	DMKGRF	DMKRGA	DMKUSO							
VMPGPND	000007	DMKCFP	DMKDSP	DMKPRV	DMKVAT								
VMPGPNT	000006	DMKCFP	DMKDSP	DMKVAT									
VMPGREAD	000010	DMKACO	DMKCKP	DMKMON	DMKPTR	DMKSCH	DMKTHI						
VMPGRINQ	000004	DMKMON	DMKPTR	DMKSCH									
VMPGWAIT	000014	DMKCFP	DMKCPU	DMKDSP	DMKPTR	DMKRPA	DMKTHI	DMKUSO	DMKVAT				
VMPGWRIT	000005	DMKACO	DMKMON	DMKPTR	DMKTHI								
VMPNCH	000006	DMKACO	DMKMON	DMKTHI	DMKVSP								
VMPNT	000061	DMKAPI	DMKBLD	DMKCFO	DMKCKP	DMKCPU	DMKCPV	DMKCQP	DMKCQR	DMKCQY	DMKDIA	DMKLOG	DMKMCT
		DMKMID	DMKMON	DMKMSG	DMKSCN	DMKTHI	DMKUSO						
VMPRGIL	000012	DMKDSP	DMKPRG	DMKPRV	DMKTMR	DMKVAT							
VMPRGPND	000003	DMKPRG											

LABEL	COUNT	REFERENCES											
VMPRIDSP	000009	DMKCFM	DMKDSP	DMKHVC	DMKSCH	DMKVCN	DMKVIC						
VMPRRCT	000006	DMKDSP	DMKSTK										
VMPSTAT	000107	DMKBLD	DMKCCW	DMKCDB	DMKCDM	DMKCDS	DMKCFG	DMKCFH	DMKCFO	DMKCFP	DMKCFS	DMKCPB	DMKCQG
		DMKCQR	DMKDEF	DMKDIA	DMKDRD	DMKDSP	DMKEXT	DMKHVD	DMKLNK	DMKLOG	DMKMON	DMKPGS	DMKPRG
		DMKPRV	DMKPTR	DMKSCH	DMKSVC	DMKTMR	DMKTRC	DMKUSO	DMKVAT	DMKVDC	DMKVMC	DMKVSI	
VMPSW	000264	DMKBLD	DMKCDB	DMKCDM	DMKCDS	DMKCFC	DMKCFG	DMKCFH	DMKCFM	DMKCFP	DMKCFS	DMKCPB	DMKDGD
		DMKDRD	DMKDSP	DMKGIO	DMKHVC	DMKHVD	DMKIOS	DMKLOG	DMKMCH	DMKMON	DMKPRG	DMKPRV	DMKPSA
		DMKSSS	DMKSVC	DMKTMR	DMKTRA	DMKTRC	DMKTRD	DMKUDR	DMKUSO	DMKVAT	DMKVCN	DMKVER	DMKVIO
		DMKVMC	DMKVSI	DMKVSP									
VMPSWAIT	000005	DMKDSP	DMKSCH	DMKTHI									
VMPSWDCA	000003	DMKALG											
VMPSWDCT	000019	DMKALG	DMKHVD	DMKJRL	DMKLNK	DMKLOG							
VMPXINT	000020	DMKCFP	DMKCPB	DMKDSP	DMKGRF	DMKPRV	DMKRGA	DMKSCH	DMKTMR	DMKVMC			
VMQBPNT	000007	DMKDSP	DMKSCH										
VMQFPNT	000019	DMKDSP	DMKSCH	DMKTHI									
VMQLEVEL	000048	DMKBLD	DMKCFO	DMKDGD	DMKDSP	DMKGIO	DMKMON	DMKPAG	DMKSCH	DMKTHI	DMKTMR	DMKUSO	
VMQPRIOR	000006	DMKMON	DMKSCH										
VMQSEND	000008	DMKDSP	DMKPSA	DMKSCH									
VMQSTAT	000026	DMKCFG	DMKCFM	DMKCPB	DMKDEF	DMKDSP	DMKGRF	DMKHVC	DMKHVD	DMKLOG	DMKQCN	DMKRGA	DMKSCH
		DMKUSO	DMKVCN	DMKVIO									
VMQ1	000016	DMKMON	DMKPAG	DMKSCH	DMKTHI	DMKTMR							
VMRBSC	000007	DMKQCN	DMKVCN										
VMRDINQ	000003	DMKMON	DMKSCH										
VMREAL	000005	DMKBLD	DMKDEF	DMKLOG	DMKPRV								
VMRON	000014	DMKCFS	DMKCQR	DMKDSP	DMKLOG	DMKPTR	DMKSCH	DMKTRA					
VMRPAGE	000006	DMKCFO	DMKPTR	DMKSCH	DMKUSO								
VMRPRIOR	000002	DMKSCH											
VMRSTAT	000231	DMKACO	DMKALG	DMKBLD	DMKCCW	DMKCDB	DMKCDM	DMKCFC	DMKCFG	DMKCFH	DMKCFM	DMKCFO	DMKCFP
		DMKCKP	DMKCNS	DMKCPI	DMKCPU	DMKCPV	DMKDGD	DMKDIA	DMKDIB	DMKDSP	DMKGIO	DMKGRF	DMKHVC
		DMKIOS	DMKLNK	DMKLOG	DMKMCH	DMKMCT	DMKMON	DMKMSG	DMKPGS	DMKPRG	DMKPRV	DMKPTR	DMKQCN
		DMKRGA	DMKRGB	DMKRNH	DMKRPA	DMKSCH	DMKSCN	DMKSVC	DMKTHI	DMKTMR	DMKTRA	DMKTRC	DMKTRD
		DMKUSO	DMKVAT	DMKVCA	DMKVCN	DMKVDD	DMKVER	DMKVIO	DMKVMA	DMKVMC	DMKVSI	DMKVSP	
VMRUN	000024	DMKDSP	DMKIOS	DMKPRV	DMKSCH	DMKTHI	DMKVCA						
VMSEG	000270	DMKAPI	DMKATS	DMKBLD	DMKCCH	DMKCCW	DMKCDB	DMKCDM	DMKCDS	DMKCFD	DMKCFG	DMKCFH	DMKCFP
		DMKCFS	DMKCKS	DMKCNS	DMKCPB	DMKCPI	DMKCPS	DMKCPU	DMKCPV	DMKCSB	DMKCSO	DMKDGD	DMKDRD
		DMKDSP	DMKERM	DMKEXT	DMKGIO	DMKGRF	DMKGRT	DMKHVC	DMKHVD	DMKIOG	DMKIOS	DMKISM	DMKLOG
		DMKMCC	DMKMCH	DMKMNI	DMKMON	DMKNLD	DMKNLE	DMKPGS	DMKPRG	DMKPRV	DMKPTR	DMKQCN	
		DMKRGA	DMKRGB	DMKRSP	DMKSCH	DMKSEP	DMKSNC	DMKSPL	DMKSSS	DMKSVC	DMKTCS	DMKTMR	DMKTRA
		DMKTRC	DMKTRD	DMKTRK	DMKUDR	DMKUDU	DMKUSO	DMKVAT	DMKVCH	DMKVCN	DMKVDR	DMKVER	DMKVIO
		DMKVMA	DMKVMC	DMKVSI	DMKVSP	DMKVSQ	DMKWRM						
VMSEGDSP	000002	DMKBLD											
VMSHADT	000008	DMKDSP	DMKVAT										
VMSHR	000027	DMKATS	DMKCCW	DMKCDB	DMKCDM	DMKCFG	DMKCPU	DMKDSP	DMKLOK	DMKPGS	DMKPRG	DMKPSA	DMKPTR
		DMKUSO	DMKVAT										
VMSHRPRC	000006	DMKATS	DMKCFG	DMKCPU	DMKLOK	DMKVMA							
VMSHRSYS	000009	DMKATS	DMKCFD	DMKCFG	DMKPGS								

LABEL	COUNT	REFERENCES											
VMSIZE	000032	DMKBLD	DMKCCW	DMKCDB	DMKCDM	DMKCDS	DMKCFD	DMKCFG	DMKCFH	DMKCFP	DMKCPI	DMKCPV	DMKDEF
		DMKHVD	DMKLOG	DMKPGS	DMKPTR	DMKUSO							
VMSLEEP	000011	DMKALG	DMKCFC	DMKCFM	DMKHVC	DMKLOG	DMKRNH						
VMSMSGON	000003	DMKMSG											
VMSPMFLG	000006	DMKCFS	DMKCQR	DMKMSG	DMKVMC								
VMSPMON	000004	DMKCQR	DMKMSG	DMKVMC									
VMSTEALS	000005	DMKMON	DMKPTR	DMKSCH									
VMSTKCNT	000015	DMKDSP	DMKSCH	DMKSTK									
VMSTKO	000010	DMKCFM	DMKCQG	DMKCQH	DMKCQP	DMKCQR	DMKCQY	DMKCST	DMKNET	DMKTHI			
VMSTMPI	000010	DMKCFS	DMKSCH										
VMSTMPT	000007	DMKSCH											
VMSTOR	000028	DMKBLD	DMKCFG	DMKCFH	DMKCFO	DMKCFP	DMKCPI	DMKCPV	DMKCQG	DMKDEF	DMKHVC	DMKLOG	DMKPGS
		DMKPTR	DMKTRD	DMKVER									
VMSVCPND	000002	DMKPRG											
VMSWPMIG	000002	DMKUSO											
VMSYSOP	000021	DMKCFM	DMKCNS	DMKCSO	DMKDSP	DMKGRF	DMKLOG	DMKLOH	DMKPSA	DMKQCN	DMKUSO	DMKVDA	DMKVDD
		DMKVDS	DMKVSQ										
VMTCDEL	000006	DMKCFT	DMKCNS	DMKCQR	DMKLOG								
VMTERM	000049	DMKACO	DMKBLD	DMKCFC	DMKCFM	DMKCFT	DMKCNS	DMKCPI	DMKCQG	DMKCQP	DMKCQR	DMKCQY	DMKDIA
		DMKHVD	DMKJRL	DMKLOG	DMKLOH	DMKMON	DMKMSW	DMKPSA	DMKQCN	DMKRGA	DMKUSO	DMKVCN	DMKVDS
VMTESCP	000005	DMKCFT	DMKCQR	DMKLOG									
VMTIMEON	000006	DMKACO	DMKCKP	DMKLOH									
VMTIMER	000018	DMKCDS	DMKCFS	DMKDSP	DMKLOG	DMKMCH	DMKPGS	DMKPTR	DMKRPA	DMKSCH	DMKTRA		
VMTIO	000007	DMKCFP	DMKVCN	DMKVIO	DMKVSI								
VMTLDEL	000005	DMKCFT	DMKCQR	DMKLOG									
VMTLEND	000016	DMKBLD	DMKCFT	DMKCNS	DMKCQR	DMKCQY	DMKGRF	DMKLOG	DMKRGA	DMKRGB			
VMTLEVEL	000054	DMKCFP	DMKCFS	DMKCQR	DMKDSP	DMKLOG	DMKPSA	DMKPTR	DMKSCH	DMKTMR	DMKTRA		
VMTMINQ	000005	DMKACO	DMKDSP	DMKSCH									
VMTMOUTQ	000040	DMKBLD	DMKCFP	DMKDSP	DMKEXT	DMKIOS	DMKLOG	DMKMCH	DMKPRG	DMKPSA	DMKSCH	DMKSVC	DMKTMR
VMTMRINT	000004	DMKPSA	DMKSCH	DMKTMR									
VMTODINQ	000004	DMKBLD	DMKSCH										
VMTON	000010	DMKCFS	DMKCQR	DMKDSP	DMKLOG	DMKPTR	DMKTRA						
VMTRBRIN	000027	DMKCDS	DMKCFC	DMKCFP	DMKDSP	DMKPRG	DMKPRV	DMKSVC	DMKTRA	DMKTRC	DMKTRD	DMKVIO	
VMTRCTL	000062	DMKCDS	DMKCFC	DMKCFP	DMKCFS	DMKDSP	DMKIOS	DMKPER	DMKPRG	DMKPRV	DMKSVC	DMKTMR	DMKTRA
		DMKTRC	DMKTRD	DMKUSO	DMKVAT	DMKVCA	DMKVIO	DMKVSI					
VMTREX	000007	DMKCFS	DMKDSP	DMKTRA	DMKTRC								
VMTREXT	000034	DMKCFM	DMKDSP	DMKPGS	DMKPRG	DMKPRV	DMKSVC	DMKTMR	DMKTRA	DMKTRC	DMKTRD	DMKUSO	DMKVIO
VMTRINT	000003	DMKTRA	DMKTRC	DMKTRD									
VMTRIO	000010	DMKDSP	DMKTRA	DMKTRC	DMKVIO								
VMTRMID	000023	DMKACO	DMKBLD	DMKCFT	DMKCQG	DMKCQP	DMKCQR	DMKCQY	DMKDIA	DMKHVD	DMKLOG	DMKLOH	DMKPSA
		DMKQCN	DMKRGB	DMKRNH	DMKUSO								
VMTRPER	000014	DMKDSP	DMKPER	DMKPRG	DMKPRV	DMKTMR	DMKTRA	DMKUSO	DMKVAT				
VMTRPRG	000005	DMKDSP	DMKPRG	DMKTRA	DMKTRC								
VMTRPRV	000008	DMKDSP	DMKPRV	DMKTRA	DMKTRC								
VMTRQBLK	000007	DMKCFS	DMKLOG	DMKSCH	DMKUSO								
VMTRSIO	000018	DMKIOS	DMKTRA	DMKTRC	DMKTRD	DMKVCA	DMKVIO	DMKVSI					

LABEL	COUNT	REFERENCES											
VMTRSVC	000005	DMKDSP	DMKSVC	DMKTRA	DMKTRC								
VMTSEND	000004	DMKDSP	DMKSCH										
VMTTIME	000021	DMKACO	DMKAPI	DMKBLD	DMKCKP	DMKCPI	DMKCPU	DMKDSP	DMKHVC	DMKLOG	DMKMON	DMKSCH	DMKTHI
		DMKTMR	DMKVSP										
VMUPRIOR	000005	DMKCFO	DMKCQR	DMKLOG	DMKMON	DMKSCH							
VMUSER	000164	DMKACO	DMKBLD	DMKCCH	DMKCDS	DMKCFO	DMKCFP	DMKCFS	DMKCKP	DMKCPE	DMKCPI	DMKCPS	DMKCPV
		DMKCQG	DMKCQH	DMKCQP	DMKCQR	DMKCQY	DMKCSE	DMKCSO	DMKCSP	DMKCST	DMKCSU	DMKCSV	DMKDEF
		DMKDGD	DMKDIA	DMKDRD	DMKHVD	DMKIOP	DMKJRL	DMKLNK	DMKLOG	DMKLOH	DMKMCC	DMKMCD	DMKMCH
		DMKMCT	DMKMIA	DMKMNI	DMKMON	DMKMSG	DMKMSW	DMKNES	DMKNET	DMKNLD	DMKNLE	DMKQCN	DMKRNH
		DMKSCN	DMKSPL	DMKSSS	DMKTHI	DMKUSO	DMKVCA	DMKVCH	DMKVDA	DMKVDC	DMKVDD	DMKVDR	DMKVER
		DMKVMC	DMKVSI	DMKVSP									
VMVCR0	000020	DMKCDB	DMKCDM	DMKCDS	DMKCFH	DMKCFP	DMKCFS	DMKDSP	DMKLOG	DMKPRV	DMKTRC	DMKVMC	DMKVSI
VMVCR14	000005	DMKCCH	DMKCFP	DMKPRV									
VMVIRCF	000020	DMKALG	DMKCFC	DMKCFM	DMKHVC	DMKLNK	DMKLOG	DMKNES	DMKNET	DMKQCN	DMKRNH		
VMVTERM	000020	DMKALG	DMKBLD	DMKCFM	DMKCFP	DMKCFT	DMKCPS	DMKDAS	DMKDEF	DMKGRF	DMKHVD	DMKLOG	DMKQCN
		DMKRGA	DMKUSO	DMKVDR	DMKVDS								
VMVTIME	000008	DMKACO	DMKLOG	DMKMON	DMKSCH	DMKTMR							
VMV370R	000074	DMKBLD	DMKCDB	DMKCDM	DMKCDS	DMKCFG	DMKCFH	DMKCFP	DMKCFS	DMKCPB	DMKCQG	DMKCQR	DMKDEF
		DMKDIA	DMKDRD	DMKDSP	DMKEXT	DMKHVD	DMKLNK	DMKLOG	DMKPRG	DMKPRV	DMKSCH	DMKSVC	DMKTMR
		DMKTRC	DMKUSO	DMKVDC	DMKVMC	DMKVSI							
VMWCNT	000008	DMKPTR	DMKRPA										
VMWNGON	000010	DMKBLD	DMKCFS	DMKCQR	DMKLOG	DMKMSG	DMKUSO						
VMWSERNG	000003	DMKSCH											
VMWSPROJ	000019	DMKBLD	DMKDSP	DMKMON	DMKSCH	DMKTHI							
VMXPG	000002	DMKPTR											
VRALOC	000003	DMKBLD	DMKDEF	DMKLOG									
VRRADD	000001	DMKCCW											
VRRBLOK	000010	DMKCCW	DMKCFP	DMKDGD	DMKGIO	DMKUNT	DMKVDS	DMKVSI					
VRRCPEX	000011	DMKCCW	DMKCFP	DMKDGD	DMKGIO	DMKUNT							
VRRRES	000008	DMKCCW	DMKCFP	DMKDGD	DMKGIO	DMKUNT	DMKVSI						
VRRSIZE	000003	DMKVDR	DMKVDS										
VRRSTAT	000008	DMKCCW	DMKCFP	DMKDGD	DMKGIO	DMKUNT	DMKVSI						
VRRUSER	000001	DMKCCW											
VSPBUFBK	000017	DMKVSP	DMKVSQ										
VSPBUFSZ	000007	DMKVSP	DMKVSQ										
VSPCAW	000014	DMKDRD	DMKVSP										
VSPCCW	000120	DMKDRD	DMKVSP										
VSPDPAGE	000025	DMKDRD	DMKVSP	DMKVSQ									
VSPIDACT	000006	DMKVSP											
VSPIDAL	000001	DMKVSP											
VSPIDASW	000008	DMKVSP											
VSPIDAW2	000002	DMKVSP											
VSPLCTL	000012	DMKCKP	DMKCSP	DMKCSQ	DMKDRD	DMKSPL	DMKVSP	DMKVSQ					
VSPNEXT	000005	DMKVSP	DMKVSQ										
VSPRECNO	000002	DMKVSP											
VSPSFBLK	000025	DMKCKP	DMKCSP	DMKCSQ	DMKDRD	DMKSPL	DMKVSP	DMKVSQ					

LABEL	COUNT	REFERENCES											
VSPSIZE	000008	DMKDRD	DMKSPL	DMKVSP	DMKVSQ								
VSPVPAGE	000016	DMKSPL	DMKVSP	DMKVSQ									
VSPXBLOK	000018	DMKCKP	DMKCQG	DMKCSP	DMKCST	DMKSPL	DMKVDR	DMKVDS					
VSPXCHAR	000005	DMKCQG	DMKCSP	DMKSPL									
VSPXCMOD	000005	DMKCQG	DMKCSP	DMKSPL									
VSPXCPYF	000004	DMKCQG	DMKCSP	DMKSPL									
VSPXDIST	000001	DMKVDS											
VSPXFCB	000005	DMKCQG	DMKCSP	DMKSPL									
VSPXFLG1	000004	DMKCQG	DMKCSP	DMKSPL									
VSPXFLSH	000004	DMKCQG	DMKCSP	DMKSPL									
VSPXLEN	000002	DMKVDR	DMKVDS										
VSPXOVLY	000005	DMKCQG	DMKCSP	DMKSPL									
VSPXSIZE	000003	DMKVDS											
VSPXTAG	000003	DMKCST	DMKSPL										
VSPXTGLN	000005	DMKCST	DMKSPL										
VSPXXUSR	000006	DMKCKP	DMKCQG	DMKCSP	DMKSPL								
VSYSRES	000003	DMKCFG	DMKCFH										
WAIT	000019	DMKAPI	DMKBLD	DMKCDS	DMKCFM	DMKCFP	DMKCPI	DMKDDR	DMKDMP	DMKDSP	DMKIOG	DMKLOG	DMKSVC
		DMKTRA	DMKTRC	DMKUSO									
WAITEND	000005	DMKDSP	DMKEXT	DMKIOS	DMKMCH	DMKPSA							
WAITSTRT	000003	DMKDSP											
WCC0	000003	DMKGRF	DMKGRT	DMKGRW									
WCC3	000010	DMKGRF	DMKGRT	DMKGRW	DMKRGA	DMKRGB							
WCC4	000005	DMKGRF	DMKGRT	DMKGRW	DMKRGA								
WCC5	000002	DMKGRF	DMKRGB										
WCC56	000002	DMKGRT	DMKRGB										
WCC6	000033	DMKGRF	DMKGRT	DMKGRW	DMKRGA	DMKRGB							
WRITBRK	000001	DMKRNH											
WRITE	000010	DMKBSC	DMKDIR	DMKLD00E	DMKLNK								
WRITEOT	000003	DMKBSC	DMKRNH										
WRITE1	000001	DMKBSC											
WRITNRM	000007	DMKRNH											
XCAPR	000002	DMKEXT											
XCDISP	000005	DMKDSP	DMKEXT										
XCMASK	000006	DMKAPI	DMKCLK	DMKDSP									
XCPEND	000019	DMKDSP	DMKEXT	DMKMCT									
XCRES	000002	DMKEXT											
XCWAK	000001	DMKEXT											
XINTBLOK	000066	DMKCFP	DMKCPB	DMKDSP	DMKGRF	DMKRGA	DMKSCH	DMKTMR	DMKVMC				
XINTCODE	000022	DMKCPB	DMKDSP	DMKGRF	DMKRGA	DMKSCH	DMKVMC						
XINTMASK	000005	DMKDSP											
XINTNEXT	000043	DMKCFP	DMKCPB	DMKDSP	DMKGRF	DMKRGA	DMKSCH	DMKTMR	DMKVMC				
XINTPARM	000002	DMKSCH	DMKTMR										
XINTSIZE	000017	DMKCFP	DMKCPB	DMKDSP	DMKGRF	DMKRGA	DMKSCH	DMKTMR	DMKVMC				
XINTSORT	000016	DMKCFP	DMKCPB	DMKDSP	DMKGRF	DMKRGA	DMKSCH	DMKTMR	DMKVMC				
XOBRCCW1	000002	DMKRSE											

LABEL	COUNT	REFERENCES											
XOBRCCW2	000001	DMKRSE											
XOBRCCW3	000001	DMKRSE											
XOBRCCW4	000001	DMKRSE											
XOBREXT	000003	DMKRSE											
XOBRFLAG	000008	DMKIOF	DMKRSE										
XOBRMIS1	000002	DMKRSE											
XOBRMIS2	000002	DMKRSE											
XOBRRT1	000006	DMKRSE											
XOBRRT2	000006	DMKRSE											
XOBRRT3	000006	DMKRSE											
XOBRRT4	000003	DMKRSE											
XOBRRT5	000007	DMKRSE											
XOBRRT6	000006	DMKRSE											
XOBRSIZE	000002	DMKRSE											
XOBRSTAT	000012	DMKRSE											
XOBRT1	000006	DMKIOF	DMKRSE										
XOBRT2	000001	DMKRSE											
XOBRT3	000004	DMKIOF	DMKRSE										
XOBR010	000003	DMKIOF	DMKRSE										
XOBR150	000004	DMKIOF	DMKRSE										
XOBR180	000002	DMKIOF	DMKRSE										
XOBR512	000008	DMKIOF	DMKRSE										
XPAGNUM	000058	DMKATS	DMKCCW	DMKCDB	DMKCDM	DMKCDS	DMKCPI	DMKDGD	DMKDRD	DMKFRE	DMKHVC	DMKHVD	DMKISM
		DMKPSA	DMKPTR	DMKQCN	DMKRPA	DMKTCS	DMKUNT	DMKVMA	DMKVMC				
XRIGHT16	000033	DMKCCW	DMKCDM	DMKCFG	DMKCPI	DMKDAS	DMKDSE	DMKERM	DMKMSG	DMKOPR	DMKRNH	DMKSEV	DMKSIX
		DMKSSP	DMKTAP	DMKTRA	DMKTRK	DMKVAT	DMKVCN	DMKVSI					
XRIGHT24	000006	DMKCCW	DMKCFP	DMKHVD	DMKSVC								
XTNDLOCK	000022	DMKDSP	DMKFRE	DMKGRF	DMKIOS	DMKLOK	DMKPAG	DMKPTR	DMKRGA	DMKVIO			
X2048BND	000016	DMKCCW	DMKCSU	DMKDRD	DMKHVD	DMKPSA	DMKTRD	DMKUNT					
X40FFS	000013	DMKCDB	DMKCDM	DMKCFG	DMKCFO	DMKCFT	DMKCPE	DMKCPS	DMKCPV	DMKMCD	DMKNLD	DMKNLE	
Y0	000010	DMKDMP	DMKDSP	DMKIOS	DMKMCH	DMKPRG	DMKPSA	DMKSVC					
Y2	000010	DMKDMP	DMKDSP	DMKIOS	DMKMCH	DMKPRG	DMKPSA	DMKSVC					
Y4	000010	DMKDMP	DMKDSP	DMKIOS	DMKMCH	DMKPRG	DMKPSA	DMKSVC					
Y6	000010	DMKDMP	DMKDSP	DMKIOS	DMKMCH	DMKPRG	DMKPSA	DMKSVC					
ZEROES	000210	DMKACO	DMKAPI	DMKBLD	DMKBSC	DMKCCH	DMKCCW	DMKCDB	DMKCDM	DMKCDS	DMKCFD	DMKCFO	DMKCFP
		DMKCFS	DMKCFT	DMKCKS	DMKCLK	DMKCPB	DMKCPI	DMKCPS	DMKCPU	DMKCPV	DMKCQG	DMKCQH	DMKCQP
		DMKCQR	DMKCQY	DMKCSB	DMKCSO	DMKCSP	DMKCSQ	DMKCST	DMKCSU	DMKCSV	DMKDAS	DMKDIA	DMKDIB
		DMKDRD	DMKDSB	DMKDSP	DMKGRF	DMKHVC	DMKHVD	DMKIOC	DMKIOE	DMKIOF	DMKLNK	DMKLOG	DMKMCD
		DMKMCH	DMKMCT	DMKMIA	DMKMID	DMKMNI	DMKMON	DMKMSW	DMKNET	DMKPSA	DMKPTR	DMKRNH	DMKRPA
		DMKRSE	DMKSCH	DMKSCN	DMKSPL	DMKSSS	DMKTAE	DMKTCS	DMKTMR	DMKTRC	DMKTRD	DMKTRK	DMKUDR
		DMKUDU	DMKUNT	DMKUSO	DMKVCA	DMKVCH	DMKVDA	DMKVDC	DMKVDR	DMKVDS	DMKVSP	DMKWRM	

CP Diagnostic Aids

This part contains the following information:

- Entry Points for CP Commands
- CP Wait State Codes
- Function Codes for DIAGNOSE Instructions

Entry Points for CP Commands

The following table is a list of CP commands and the modules that gain control to perform their functions.

Command	Entry Label
ACNT	DMKCPVAC
ADSTOP	DMKCFDAD
ATTACH	DMKVDAAT
ATTN	DMKCFCRQ
AUTOLOG	DMKALGON
BACKSPAC	DMKCSOBS
BEGIN	DMKCFCBE
CHANGE	DMKCSUCH
CLOSE	DMKCSPCL
COUPLE	DMKDIACP
CP	DMKCFM
DCP	DMKCDBDC
DEFINE	DMKDEFIN
DETACH	DMKVDDDE
DIAL	DMKDIAL
DISABLE	DMKCPVDS
DISCONN	DMKUSODS
DISPLAY	DMKCDBDI
DMCP	DMKCDMDM
DRAIN	DMKCSODR
DUMP	DMKCDMDU
ECHO	DMKMSGEC
ENABLE	DMKCPVEN
EXTERNAL	DMKCPBEX
FLUSH	DMKCSOFL
FORCE	DMKUSOFL
FREE	DMKCSPFR
HALT	DMKCPSH
HOLD	DMKCSPHL
INDICATE	DMKTHIEN
IPL	DMKCFGIP
LINK	DMKLNKIN
LOADBUF	DMKCSBLD
LOADVFCB	DMKCSBVL
LOCATE	DMKCFDLO
LOCK	DMKCPVLK
LOGOFF	DMKUSOLG
LOGON	DMKLOGON
MESSAGE	DMKMSGMS
MONITOR	DMKMCCCL
NETWORK	DMKNETWK
NOTREADY	DMKCPBNR
ORDER	DMKCSUOR
PURGE	DMKCSUPU
QUERY[1]	DMKCFCQU
READY	DMKCPBRY
REPEAT	DMKCSORP
REQUEST	DMKCFCRQ
RESET	DMKCPBRS
REWIND	DMKCPBRW
SAVESYS	DMKCFHSV
SET[1]	DMKCFCSE
SHUTDOWN	DMKCPSSH
SLEEP	DMKCFCSL
SMSG	DMKMSGSM
SPACE	DMKCSOSP
SPOOL	DMKCSPSP
START	DMKCSOST
STCP	DMKCDSCP
STORE	DMKCDSTO
SYSTEM	DMKCPBSR
TAG	DMKCSTAG
TERMINAL	DMKCFTRM
TRACE	DMKTRACE
TRANSFER	DMKCSUTR
UNLOCK	DMKCPVUL
VARY	DMKCPSRY
WARNING	DMKMSGWN
*	DMKCFM

[1]Major operand decode of QUERY and SET is by a scan table in DMKCFMQU. Depending on the operand match, DMKCQP, DMKCQG, DMKJRL, or DMKCQR is called for QUERY. The respective entry points are DMKCQPRV, DMKCQGEN, DMKJRLQU, and DMKCQREY. For SET, DMKCFS, DMKJRL, or DMKCFO is called. Respective entry points are DMKCFSET, DMKJRLSE, and DMKCFOEX.

Figure 25. CP Commands and Their Module Entry Points

CP Wait State Codes

A wait state is produced by one of the following modules:

DMKCCH	DMKMCH
DMKCKP	DMKPAG
DMKCPI	DMKSAV
DMKDMP	DMKWRM

When a wait state occurs, the Program Status Word (PSW) is displayed at the operator's console in the following format:

```
xxyyyyyyzzzzzwww
```

where:

xxyyyyyy is the left half of the program status word. This half may be either:

03yyyyyy Valid wait condition. The system is waiting for work.

00yyyyyy System wait caused by an error condition.

zzzzzwww is the right half of the program status word. The wait state code is found in the right half of the PSW when the CPU is in the wait state. The wait state code, www, indicates the error condition.

Wait Code	Explanation
001	The machine check handler found an unrecoverable failure. Probable hardware error.
002	The channel check handler found an unrecoverable failure. Probable hardware error.
003	A system failure occurred before a valid warm start was performed.
004	This wait state code is loaded by DMKDMP when a console, or an output device is not operational, or when a console or output device produces an inexplicable error status. Probable hardware error.
005	DMKCPI could not find an operational primary or alternate console. Probable hardware error.
006	This is a normal wait when a system shutdown is completed.
007	A program check, a machine check, or a permanent I/O error was found by the checkpoint program.
008	Checkpoint and system shutdown are complete. If the system is running under an alternate console, error messages DMKCKP910I, DMKCKP911W, DMKCKP960I, and DMKCKP961I are not displayed.
009	An error condition occurred that prevents a warm start. If the system is running under an alternate console, error messages DMKCKP910I and DMKCKP911W are not displayed.
00A	A machine check occurred while DMKSAV was attempting to save or restore a page image copy of the nucleus on a SYSRES device. Probable hardware error.
00B	A machine check occurred before initialization was complete.
00C	An attempt was made to IPL from a disk that did not contain a system. Thus, the wait state code 00C entered on disk by the Format/Allocate program is encountered.
00D	The machine size defined during system generation is greater than the real machine size, or a hardware error has occurred which inhibits VM/370 from using the required storage.

00F Hardware errors are being received on VM/370 paging device(s). The wait state that causes this code is preceded by message

DMKPAG415E CONTINUOUS PAGING ERRORS FROM DASDxxx

010 The SYSRES device, on which DMKSAV is attempting to write a page image copy of the nucleus, is not mounted or not ready.

011 An unrecoverable error, other than a machine check, occurred while DMKSAV attempted to write a page image copy of the nucleus on the SYSRES device.

012 The normal wait state code loaded by DMKSAV when it has completed loading the nucleus.

013 The machine check handler encountered an unrecoverable error on the attached processor. Probable hardware error.

015 A SIGP issued to the attached processor during system initialization by DMKCPI or DMKAPI was unsuccessful.

027 An unrecoverable I/O error occurred or system input is incorrect.

CP Abend Codes

The CP abends, their causes and required actions are listed in the IBM VM/370 System Messages.

Function Codes for DIAGNOSE Instructions

Figure 26 indicates the DIAGNOSE codes used in VM/370 and gives a brief explanation of their uses.

Function Code	Class	Function	DMKHVC Label	DMKHVD Label
000	G	Store extended identification code.		HVDSTIDX
004	C,E	Examine data from real storage.		READCPC
008	G	Execute VM/370 CP command.	HVCONFN	
00C	G	Pseudo-timer facility.	HVCHRON	
010	G	Release virtual storage pages.	HVCPGRL	
014	G	Manipulate input spool files.		HCDSPRD
018	G	Standard DASD I/O.	HVCDISK	
01C	F	Clear error recording area.		HVDLRER
020	G	General virtual I/O interruptions.	HVCFAKE	
024	G	Virtual device type inquiry.		HVDDTYP
028	G	Dynamic TIC modification.	HVCDCPM	
02C	C,E,F	Get DASD address of error recording and number of cylinders allocated for error recording.		HVDEREP1
030	C,E,F	Read a page of error recording data.		HVDEREP2

Figure 26. Function Codes for DIAGNOSE Instruction (Part 1 of 2)

Function Code	Class	Function	DMKHVC Module	DMKHVD Module
034	C,F	Reads the system dump spool file.		HVDRSDF
038	C,E	Reads the system symbol table.		HVDRDSYM
03C	A,B,C	Dynamically updates the VM/370 directory.		HVDDIRCT
040		Reserved for IBM use.	HVCEXIT	
044		Reserved for IBM use.	HVCEXIT	
048		Reserved for IBM use.	HVCEXIT	
04C	any	Generate accounting cards.		HVDACCT
050	A,B,C	Saves 3704/3705 control program image.		HVD3705
054		Enable or disable external interruptions.		HVDEXPA
058	G	Virtual console interface for 3270.	HVCGRAF	
05C		Edit message according to EMSG settings.	HVCEMSG	
060		Provide virtual machine storage size.	HVCSTOR HVCSTOR	
064		Load, find, or purge a named system.	HVCSYS	
068	G	Virtual Machine Communication Facility.	HVCVMCF	
074	A,B,C	Loads a 3800 named system into virtual storage		HVD3800
078	any	MSS communication	HVCSSS	
084	B	Updates in-place a VM/370 directory control statement in its online control block form.	DMKUDU	
100		Start of functions specified by a user.	HVCUSER	

Figure 26. Function Codes for DIAGNOSE Instruction (Part 2 of 2)

Appendix A: VM/370 Extended Control-Program Support

VM/370 Extended Control-Program Support (ECPS)

VM/370 Extended Control-Program Support (ECPS) consists of three hardware-assisted parts:

1. Control program assist (CP assist) - defines new hardware instructions to assist CP routines and functions or, in two cases, as new interpretations of existing VM/370 instructions. CP assist does not operate in a VM/370 system that runs under VM/370.

2. Expanded virtual machine assist - provides an expansion of the existing virtual machine assist.

3. Virtual interval timer assist - provides a more accurate hardware updating of the interval timer for the virtual machine.

See VM/370 Planning and System Generation Guide for a list of the processors on which ECPS is available.

ECPS INTERACTION WITH OTHER FUNCTIONS

- Virtual machine assist - The expanded virtual machine assist can be enabled only if virtual machine assist is also enabled.

- Program event recording - No PER events are recognized by CP assist. Virtual machine assist does recognize PER events for certain instructions. PER events are not recognized during the updating of the virtual interval timer.

- VS1 assist - VM/370 ECPS and VS1 assist do not interfere with each other.

- DOS emulator - If the DOS emulator is active, virtual machine assist is disabled. CP assist and the virtual interval timer assist are not disabled if the DOS emulator is active.

CONTROL BY CONTROL REGISTER 6 AND MICBLOK ASSIST CONTROL FIELD

The contents of control register 6 exercise overall and absolute control over virtual machine assist, CP assist, expanded virtual machine assist, and virtual interval timer assist. Values in control register 6 share control of functions provided by expanded virtual machine assist with the setting of bits in the MICBLOK's assist control field. The use of the assist control field is described later on under the topic "Expanded Virtual Machine Assist."

The following table defines the contents of control register 6:

Bit	Description
0	Virtual machine assist enabled if on, disabled if off
1	Virtual machine in problem state if on, in supervisor state if off

2	ISK and SSK instructions not allowed if on, allowed if off
3	System/360 instructions only if on, System/370 instructions if off
4	Virtual SVC interrupts not allowed if on, allowed if off
5	Shadow table fixup allowed if on, not allowed if off
6	Control program assist enabled if on, disabled if off
7	Virtual interval timer support enabled if on, disabled if off
8-28	Real address of virtual machine pointer list
29-31	Unused, must be zero

Summary of Hardware Assist Control

The following chart summarizes the control by control register 6 of virtual machine assist, extended virtual machine assist, CP assist, and the virtual interval timer assist:

Virtual Machine Assist Bit 0	Control Program Assist Bit 6	Virtual Interval Timer Assist Bit 7	Enabled Assists	System Operator Command	User Command
0	0	0	None	B, D	
0	0	1	None	*	
0	1	0	CP assist	B, C	
0	1	1	CP assist	*	
1	0	0	Virtual machine assist	A, D	E, NOTMR
1	0	1	Virtual machine assist, virtual interval timer assist	A, D	E, TMR
1	1	0	Virtual machine assist, CP assist, expanded virtual machine assist	A, C	E, NOTMR
1	1	1	Virtual machine assist, CP assist, expanded virtual machine assist, virtual interval timer assist	A, C	E, TMR

* Not possible with VM/370
A indicates SET SASSIST ON
B indicates SET SASSIST OFF
C indicates SET CPASSIST ON
D indicates SET CPASSIST OFF
E indicates SET ASSIST ON with TMR or NOTMR as indicated

VIRTUAL MACHINE POINTER LIST

The virtual machine pointer list (MICBLOK) is a 24-byte area that starts on a doubleword boundary and contains eight fullwords. Control register 6 contains an address that points to the beginning of the virtual machine pointer list. The address is formed by concatenating bits 8 through 28 of control register 6 with three low-order zero bits. All storage accesses to the virtual machine pointer list are done with real addresses and with a storage protect key of 0.

The following table defines the fullwords at the specified offsets into the virtual machine pointer list. The usage of the workspace pointer and the workspace itself is implementation-dependent. The usage of the pointer and the workspace is not further defined in this document.

Offset	Definition
+0	Real segment table pointer
+4	Virtual control register pointer
+8	Virtual PSW pointer
+12	Workspace pointer
+16	Virtual interval timer pointer
+20	Assist control field
+24	Reserved for IBM use

TRACE TABLE ENTRIES

The first bit of each VM/370 trace table entry generated by ECPS is set to one (1). Information about VM/370 trace table entries is contained in the IBM VM/370 System Programmer's Guide.

RELATIONSHIPS BETWEEN HARDWARE ASSISTS

Figure 27 illustrates the possible ways of running a virtual machine with various combinations of hardware assists and how the SET command affects their operation.

Figure 27. Hardware Assist Relationships (Part 1 of 4)

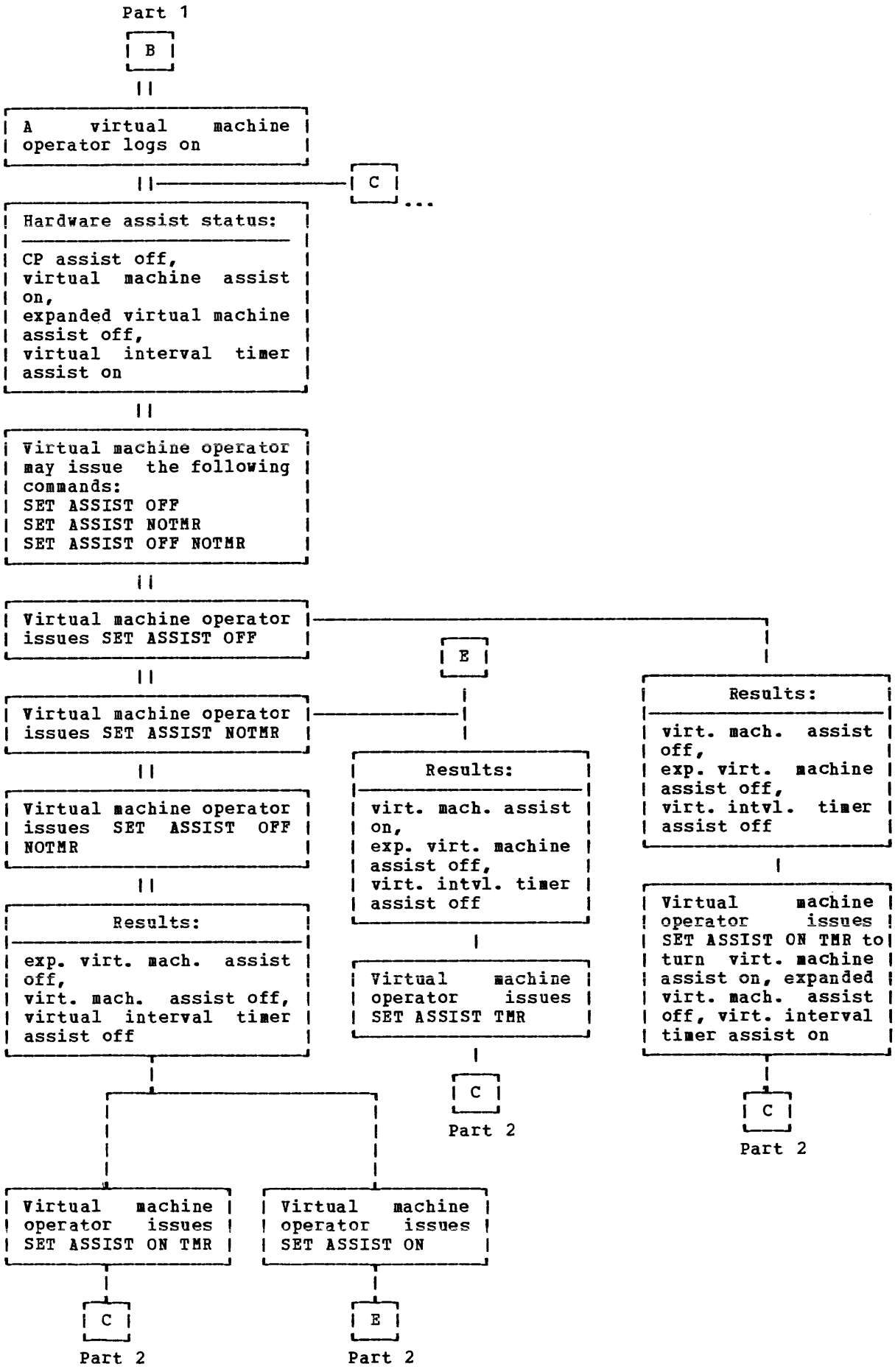


Figure 27. Hardware Assist Relationships (Part 2 of 4)

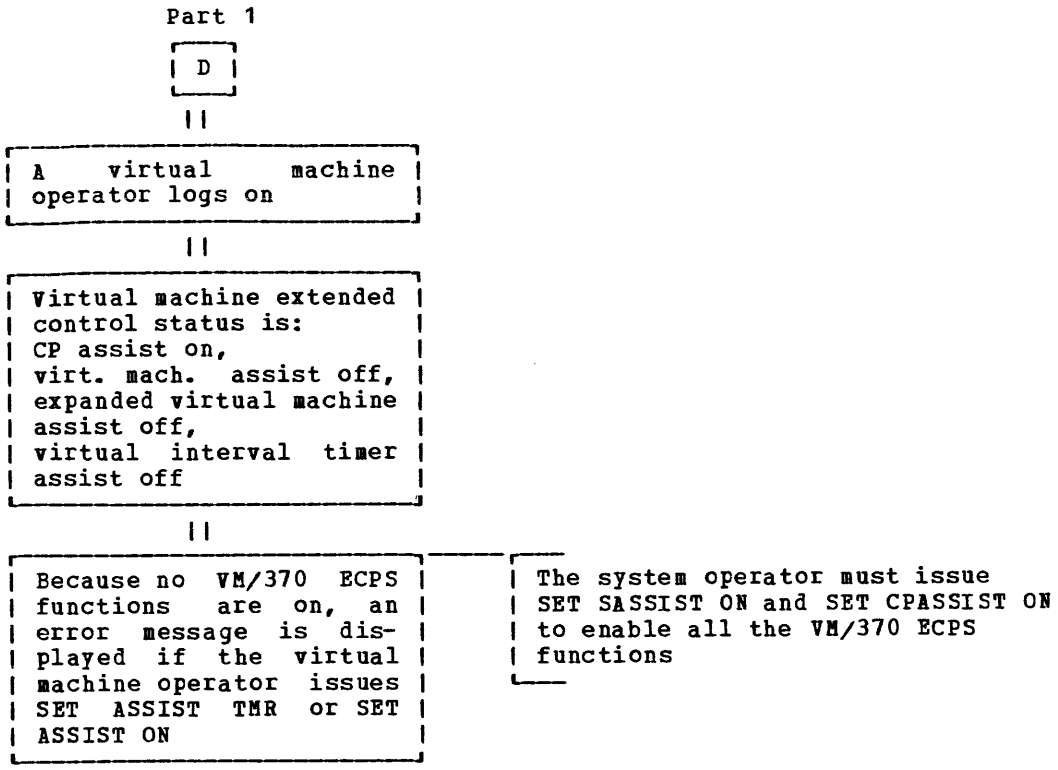


Figure 27. Hardware Assist Relationships (Part 3 of 4)

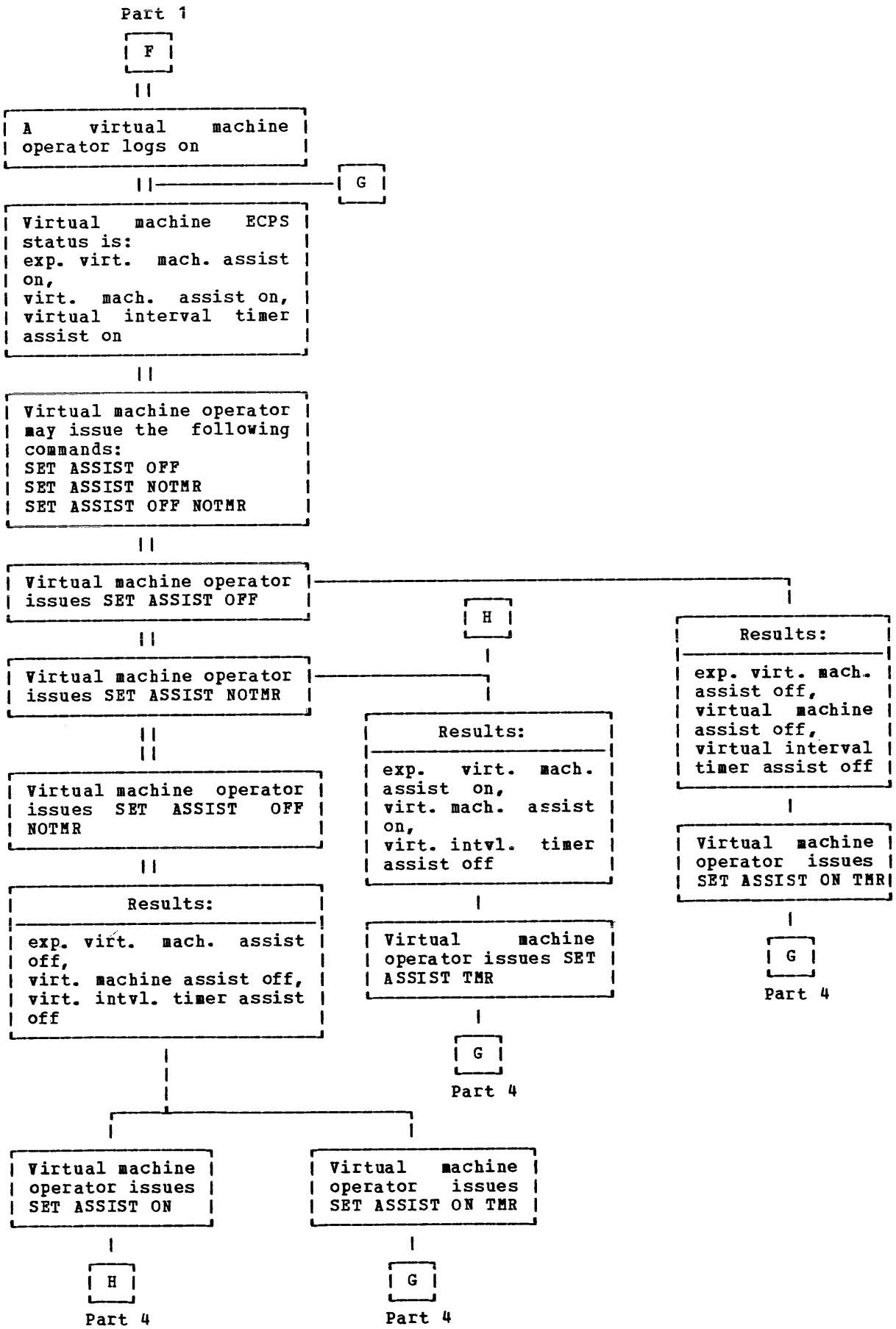


Figure 27. Hardware Assist Relationships (Part 4 of 4)

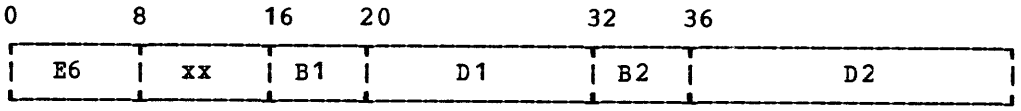
CONTROL PROGRAM ASSIST (CP ASSIST)

The following information describes the CP assist functions that are invoked directly or indirectly by the VM/370 Control Program.

CP assist is part of ECPS. CP assist is hardware that assists VM/370 Control Program functions with new instructions.

Instruction Format of CP Assist Instructions

Except for the functions initiated by VM/370 SVCs, all CP assist functions are System/370 instructions. The CP assist instructions are six bytes long and are in an SS format. The format is:



where:

E6 is the first byte of the op code for all CP assist instructions

xx is the second byte of the op code and is an op code extension that is unique for each instruction.

Each instruction has two operands that are effective addresses calculated from the B1D1 and B2D2 values in the normal System/370 way. The specific CP assist instruction determines whether these operands are data, addresses, or not used.

Instruction Summary for Control Program Assist

The following table summarizes the operation codes, instruction names, and descriptions of the CP assist instructions:

Op Code	Trace Table Entry	Instruction	VM/370 CP Function
E600	Yes	FREE	Get free storage space
E601	Yes	FRET	Return free storage space
E602	No	PTRLK	Lock a page
E603	No	PTRUL	Unlock a page
E604	No	DECCW0	Decode subsequent CCW commands
E605	No	UNTFR	Free CCW storage
E606	No	SCNVU	Locate virtual I/O control blocks
E607	No	DSP1	Dispatch a block or a virtual machine - full function
E608	No	TRANBRNG	Test page status
E609	No	TRANLOCK	Test page status and lock
E60A	No	ZAPSEGS	Invalidate segment table
E60B	No	ZAPPAGE	Invalidate page table
E60C	No	DECCW1	Decode first CCW command
E60D	No	DSP0	Main entry to dispatcher
E60E	No	SCNRU	Locate real I/O control blocks
E60F	No	CCWGENRL	Common CCW command processing
E610	No	UNTRN	Untranslate CSW

Op Code	Trace Table Entry	Instruction	VM/370 CP Function
E611	Yes	DSP2	Dispatch a block or a virtual machine - abbreviated function
E612	No	STECPSVM	Store VM/370 ECPS identification
E613	No	SHARED	Locate changed shared page
0A08	Yes	LINK	VM/370 Control Program SVC LINK
0A0C	Yes	RETURN	VM/370 Control Program SVC RETURN

Locating CP Assist Instructions in Assembly Listings

The following example shows how to identify CP assist instructions in assembly listings.

```
                     USING DMKFREE,R15                        @V386918 00218700
               DMKFREE DS                                     @V386198 00218800
                       DS                                     @V386198 00219200
 E600F718F64B          DC    X'E600',S(SUBTABLE,BYTBL-1)  ***V386198 00219300
 ..                    STM   R0,R15,FREESAVE        .........%V386198 00220000
 .                     LR    R12,R15                .        %V386198 00221000
-.-------------------------------------------------------.-------------------
 .                                                       .
 .... CP assist instruction is                           .. CP assist function exe-
      identified by E6 op code                              cutes software instruc-
                                                            tions identified by (%)
                                                            sign.
```

Note: Instructions marked with % in column 64 may not actually be executed if CP assist is enabled. For example, register contents may not be stored in temporary save areas and certain additional constants may be referenced. However, the functional equivalent of the group of instructions marked by the % is performed by CP assist.

Locating X'E6' Instructions Using DMKCPI and a Load Map

DMKCPI ENTRY POINT LIST: In the module DMKCPI is a list of entry points that contain the control program assist instructions (X'E6' op code). This list has the label CPATABLE. This list is shown in the following example:

```
                    Entry          CP Assist
     Label          Point          Op Code

     CPATABLE DS    0F
              DC    V(DMKFREE)     E600
              DC    V(DMKFRET)     E601
              DC    V(DMKPTRLK)    E602
              DC    V(DMKPTRUL)    E603
              DC    V(DMKCCW0)     E604
              DC    V(DMKUNTFR)    E605
              DC    V(DMKSCNVU)    E606
              DC    V(DMKDSP1)     E607
              DC    V(DMKCCWB1)    E608
              DC    V(DMKCCWB2)    E608
              DC    V(DMKCCWB3)    E608
```

```
Label                Entry Point          CP Assist Op Code

          DC     V(DMKCCWB4)          E608
          DC     V(DMKCCWB5)          E608
          DC     V(DMKCCWB6)          E608
          DC     V(DMKCCWB7)          E608
          DC     V(DMKCCWB8)          E608
          DC     V(DMKCCWL1)          E609
          DC     V(DMKCCWL2)          E609
          DC     V(DMKCCWL3)          E609
          DC     V(DMKCCWL4)          E609
          DC     V(DMKCCWL5)          E609
          DC     V(DMKVATZS)          E60A
          DC     V(DMKVATZP)          E60B
          DC     V(DMKCCW1)           E60C
          DC     V(DMKDSP0)           E60D
          DC     V(DMKSCNRU)          E60E
          DC     V(DMKCCWGN)          E60F
          DC     V(DMKUNTRN)          E610
          DC     V(DMKDSP2)           E611
```

LOAD LIST FOR ENTRY POINT LOCATIONS: The following is an example of the load list for module DMKFRE that contains the entry points DMKFREE and DMKFRET. If you consult the DMKCPI entry point list above, you will find that entry points DMKFREE and DMKFRET contain the CP assist operation codes X'E600' and X'E601'.

```
:READ  DMKFRE   TEXT     xx  xx-xxx  5/14/76   11:15
                         ****************************************
                         *                                      *
                         * This area contains a list of updates *
 ┌─┐                     * and macro  libraries  included in the *
 |1|.....                * assembly.                           *
 └─┘     .               *                                      *
         .               ****************************************
 ┌─┐     DMKFRE   AT 01A740
 |2|....DMKFREE  AT 01A740
 └─┘  ...DMKFRET  AT 01AB38   ┌─┐
    .   DMKFRETR AT 01AB26    |1| This is the module DMKFRE loaded at
 ┌─┐ .   DMKFRELG AT 01AF20   └─┘ address 01A740.
 |3|..   DMKFRELS AT 01AE48
 └─┘     DMKFRELO AT 01AE54   ┌─┐
         DMKFREHI AT 01AD88   |2| This is the entry point DMKFREE loaded
         DMKFRENP AT 01AE8C   └─┘ at address 01A740. The entry point
         DMKFRESV AT 01AE9C       list in DMKCPI shows that this entry
         DMKFREST AT 01AE58       point contains an X'E600' op code.
         DMKFRETL AT 01AD80
         DMKFRERS AT 01AAC8   ┌─┐
                              |3| This is the entry point DMKFRET loaded
                              └─┘ at address 01AB38. The entry point
                                  list in DMKCPI shows that this entry
                                  point contains an X'E601' op code.
```

Turning Off Selected Hardware Assisted Functions

The following information describes how to turn off selected ECPS functions if you suspect a problem with the function.

In some cases, hardware-assisted functions are invoked by other hardware-assisted functions and they must all be made NOOPs to completely disable the function. To find all the functions that are

related and that must all be turned off to disable the function, see the following instruction cross-reference list.

Those hardware-assisted functions that are not in the list may be turned off individually without being concerned about other functions that may call it.

The SVC8 and SVC12 functions of CP assist can be disabled by setting certain input parameters in the VMALIST to extreme values. The various functions provided by expanded virtual machine assist are controlled by the setting of bits in the MICBLOK's assist control field, see the topic "Expanded Virtual Machine Assist" for details.

Instruction/Function Cross-Reference List

Function	Invoked By
DECCW1	DECCW0
DSP2	DSP1
FRET	UNTFR, DSP2
PTRLK	CCWGENRL, TRANLOCK
PTRUL	UNTFR
RETURN	CCWGENRL, UNTFR
SHARED	DSP2
TRANBRNG	DECCW1
ZAPPAGE	ZAPSEGS
ZAPSEGS	DSP2, PTLB

Note: Be careful when making these functions NOOPs. For example, to completely turn off ZAPPAGE, ZAPPAGE and ZAPSEGS must be made NOOPs. But ZAPSEGS is invoked by DSP2 and PTLB. Therefore, DSP2 and PTLB must both be made NOOPs. DSP2 is invoked by DSP1; therefore, DSP1 must also be made a NOOP. The end result must be that DSP1, DSP2, PTLB, ZAPZEGS and ZAPPAGE must all be turned off.

| Note: DSP0, DSP1 and DSP2 CP ASSIST instructions are changed to NO-OP
| instructions in System Extensions Program Product, Program No. 5748-XE1.

Using the Instruction/Function Cross-Reference List To Defeat Functions

As an example, assume that you want to defeat the FRET hardware-assisted function.

1. Look up FRET in the Instruction/Function Cross Reference List. FRET is invoked by UNTFR and DSP2. DSP2 is invoked by DSP1. Therefore, the four functions that must be deleted are: FRET, UNTFR, DSP2, and DSP1.

2. FRET is part of DMKFRE, UNTFR is part of DMKUNT, DSP2 and DSP1 are part of DMKDSP. These are the modules that contain the hardware assisted code that must be made NOOPs.

3. Listed in the CPATABLE list of DMKCPI are the following entry points:

   DMKFRET which contains the CP assist instruction E601
   DMKUNTFR which contains the CP assist instruction E605
   DMKDSP1 which contains the CP assist instruction E607
   DMKDSP2 which contains the CP assist instruction E611

4. The load map for your VM/370 system lists the modules and entry points, and their locations.

5. Using the DMKCPI list to find the CP assist op code and entry point name, and the load list to find the location of that entry point, replace the CP assist instructions (X'E6xxxxxxxxx') in those locations with three NOOPs (X'470000000700').

Exceptions to Control Program Assist

All CP assist instructions cause operation exceptions if ECPS is not installed. The CP assist instructions are executed as NOOPs if the PSW is in supervisor state and VM/370 ECPS is installed but the CP assist is disabled in control register 6. If VM/370 ECPS is installed but the PSW is in a problem state, the CP assist instructions cause privileged operation exceptions regardless of the setting of control register 6 bit 6.

EXPANDED VIRTUAL MACHINE ASSIST

Expanded virtual machine assist is part of the VM/370 Extended Control-Program Support. Expanded virtual machine assist is hardware that executes certain privileged operations issued by a VM/370 virtual machine. Expanded virtual machine assist is an extension of the Virtual Machine Assist feature. Expanded virtual machine assist is invoked only if virtual machine assist cannot execute the subject instruction in the virtual machine, a corresponding virtual machine assist function is defined, and that virtual machine assist function would present a privileged operation exception. In some cases, expanded virtual machine assist does not completely execute the subject instruction.

Instruction Summary for Expanded Virtual Machine Assist

The following table summarizes the operation codes, instruction names, amount of execution of the instruction by expanded virtual machine assist, and descriptions of the instructions:

Op Code	Trace Table Entry	Instruction	Amount of Execution of VM/370 Function	Virtual Machine Definition
82	Yes	LPSW	Partial	Load PSW
B20D	No	PTLB	Complete	Purge Table Lookaside Buffer
B206	Yes	SCKC	Partial	Set Clock Comparator
9C	Yes	SIO,SIOF	Partial	Start I/O, Start I/O Fast
B208	Yes	SPT	Partial	Set CPU Timer
80	Yes	SSM	Partial	Set System Mask
AC	Yes	STNSM	Partial	Store Then AND System Mask
AD	Yes	STOSM	Partial	Store Then OR System Mask
B209	No	STPT	Complete	Store CPU Timer
9F	No	TCH	Complete	Test Channel

Enabling and Disabling Expanded Virtual Machine Assist Functions

Although all of the functions provided by expanded virtual machine assist can be disabled by values in control register 6, any of the functions of expanded virtual machine assist can be enabled or disabled by the setting of bits in a field defined in the MICBLOK (displacement X'14', label MICEVMA) in conjunction with values in control register 6 that enable expanded virtual machine assist. Specifically, if bits 0 and 6 of control register 6 are both on and the real machine is in virtual supervisor state, then the finer controls provided in the specified MICBLOK field are active. If those two bits in control register 6 are not both on, or if the real machine is not in virtual supervisor state, then the settings of the MICBLOK assist controls are ignored.

During virtual supervisor state execution, a particular function of expanded virtual machine assist is enabled only if:

- Expanded virtual machine assist is enabled in control register 6 (bits 0 and 6 are both on), and
- The bit defined for that function in the MICBLOK assist control field is on.

Virtual machine assist is unaffected by the MICBLOK assist control field.

Some bits in the MICBLOK assist control field enable and disable several functions of expanded virtual machine assist. This is because of the similarity in the definitions of the controlled functions. The following table contains the bit definitions of the MICBLOK assist control field.

Bit	Functions
0	Load PSW
1	Purge TLB
2	Set clock comparator, set CPU timer
3	Start I/O
4	Set system mask, store then AND system mask, store then OR system mask
5	Store CPU timer
6	Test channel

These MICBLOK assist control bits can also be used to enable or disable all of expanded virtual machine assist. If bits 0 and 6 of control register 6 are both on, and the assist control bits are off, then CP assist is enabled but expanded virtual machine assist is disabled.

VIRTUAL INTERVAL TIMER ASSIST

Virtual interval timer assist is the hardware support for an interval timer in the virtual machine. The virtual machine may be in either EC mode or BC mode. The virtual interval timer is in the virtual machine's page frame 0 and functions like the real interval timer.

The virtual interval timer runs without either the CP assist or the extended virtual machine assist being active. The virtual interval timer assist is only active if virtual machine assist is active. Bits 0 and 7 of control register 6 control the virtual interval timer assist.

Virtual Interval Timer Maintenance

Bit 23 of the virtual interval timer is decremented like the real interval timer. If a virtual machine is executing (the real PSW is in problem state) and the real interval timer needs updating, both the real interval timer and the virtual interval timer are updated. The virtual interval timer is decremented only if the following conditions are true:

- The real PSW is in the problem state (bit 15 = 1)
- Virtual machine assist is enabled (control register 6 bit 0 = 1)
- Virtual interval timer assist is enabled (control register 6 bit 7 = 1)

Virtual Interval Timer Assist Interrupt

A virtual interval timer interrupt request is recognized when the virtual interval timer is decremented from a positive value, including zero, to a negative value. When this condition occurs, an interval timer interrupt is presented to the virtual machine.

If the virtual interval timer interrupt cannot be presented to the virtual machine, the virtual interval timer interrupt is presented to the real machine. When presented to the real machine, the virtual interval timer interrupt is distinguished from the real interval timer interrupt by a unique external interrupt code (bit 7 of the halfword interrupt code set to 1 and bits 0-6 set to 0). Bits 8-15 of the interrupt code are zero unless set to one for another condition that may be concurrently indicated. (Locations X'84' and X'85' are set to zero.)

VM/370 MSS Support

Following are annotated flow diagrams for the logic to support the IBM 3850 Mass Storage System.

LOGON A USER HAVING A MINIDISK ON AN UNMOUNTED SYSTEM VOLUME

DMKLNK, CHK3330V
A required system volume is not mounted, try to get a 3330V mounted if the minidisk is a 3330.

DMKSSSLN
Entry to mount an MSS system volume.

DMKSSS, FINDRDEV
Allocate a SYSVIRT real 3330V device. This may involve demounting a volume which is mounted but not in use. If there are none such volumes available, issue message DMKSSS080E and return with return code 8.

DMKSSS, BLDCOMMT
Construct an MSSCOM, filling in the volume serial, device address selected, type of request (mount), and userid.

DMKSSS, SETMNTFG
Build a CPEXBLOK for the return to DMKLNK after the MSC has processed the request. Chain it from field MSSTASK2. Build a CPEXBLOK for the return to DMKLNK after the mount is complete (pack change interruption received on the 3330V). Chain it from field MSSTASK1.

DMKSSS, SCHMSSC
Put the MSSCOM in the queue, generate an attention interruption for the communication device if necessary, and exit to DMKDSP.

DMKSSS, HVC04ENT
Entry when DIAGNOSE code X'78', subcode 4 is received. OS/VS is ready to process an MSC request. Place the next MSSCOM in the virtual machine, and return to DMKHVC.

DMKSSS, HVC08ENT
Entry from DMKHVC when DIAGNOSE code X'78', subcode 8 is received. The MSC has processed the mount request.

DMKSSS, RESETMQR
If there was an MSS error, write message DMKSSS083E and return to DMKLNK with return code 8.

DMKSSS, MNTCOM
If there was not an MSS error, indicate that the MSSCOM is now waiting for the pack change interruption. Write message DMKSSS088I. Return to DMKLINK with return code 4.

DMKLNK, MNTSETUP
Return from DMKSSS. Save the current workarea and control information. Return to caller.

| DMKDSB
| Entry from DMKDAS on pack change interruption. If the device is a 3330V, look for an MSSCOM waiting for this volume serial. If one is found, stack a CPFXBLOK for entry to DMKSSSEN. Exit to DMKDSP.

| DMKSSSEN
| Pick up the CPEXBLOK for DMKLNKSS and stack it.

| DMKLNKSS
| Complete the LINK processing for the minidisk.

LOGON A USER HAVING A 3330V DEDICATED AS A 3330V

| DMKLOG, CALLMSSA
| Determine that a virtual 3330V is needed, save the UDEVBLOK, call DMKSSSL1.

| DMKSSSL1
| Go through device allocation, etc., to schedule a mount.

| DMKLOG, MSSMOUNT
| If an MSS mount is in process (return code 4 from DMKSSS), proceed to get the next directory statement. Otherwise, find the RDEVBLOK for the device that DMKSSS allocated and continue the dedicat process.

| DMKLOGSS
| Entry from DMKDSB and DMKSSSEN after mount.

| DMKSCNRU
| Get the RDEVBLOK

| DMKVDSAT
| Attach the virtual device.

| DMKLOG, TSTV333V
| If the virtual device is a 3330V, set flag RDEV333V to indicate that there is no CP MSS CCW prefix.

| DMKLOG, FREEUDEV
| If there is virtual I/O waiting, as indicated by a CPEXBLOK address in field MSSTASK3 of the MSSCOM used for the mount, stack the IOBLOK. Return to DMKDSP.

PROCESS DIAGNOSE CODE X'78'

| DMKSSSHV
| Entry from DMKHVC when DIAGNOSE code is X'78'.

| DMKSSS, HVC00ENT
| The entry subcode was 0. Save the communication device address and the communicator VMBLOK address. Set PSAMSS indicating that the MSC is now available.

| DMKSSS, HVC04ENT
| The entry subcode was 4. If there is an MSSCOM in the queue to be processed, call DMKPTRAN to get the communicator's buffer address. Put the MSSCOM in the virtual machine buffer.

| DMKSSS, HVC08ENT
| The entry subcode was 8. The MSC has processed a request. If there was an error, write message DMKSSS088E, dequeue the MSSCOM, stack the return to the DMKSSS caller from MSSTASK2 with a return code 8, and return to DMKHVC. If there was no MSC error, stack the MSSTASK2 CPEXBLOK with a return code of 4, and return to DMKHVC.

| GENERATE THE CHANNEL PROGRAM PREFIX FOR A 3330V

| DMKCCW
| Entry to generate a real channel program from a virtual machine channel program.

| DMKCCW, CCWINDSD
| If the real device is a 3330V, set a flag indicating that the MSS channel program prefix is needed.

| DMKCCW, CCW02
| If the prefix-needed flag is on and the virtual device is not a virtual 3330V, put the prefix in the RCWTASK.

| DMKCCW, DASDTBL AND DEDDTBL
| These are tables of addresses of routines that are to get control to process specific CCW operation codes for DASD and dedicated devices. In each subroutine, a check is made to see if there is an unresolved MSS prefix. If so, it checks to see if the virtual channel program contains a SEEK. If so, it checks to see if the argument is used to generate the SEEK argument for the prefix. If not, the prefix CCW is set to SEEK to cylinder 0.

| GENERATE THE CHANNEL PROGRAM PREFIX FOR CMS I/O TO A 3330V

| DMKDGD
| Entry to process I/O requests to DASD as initiated by the special DIAGNOSE code '78' interface from CMS.

| DMKDGD, NOPRE
| If the real device is a 3330V, set up the prefix in the RCWTASK.

| DMKDGD, CHKMOUNT
| The VDEVBLOK for the virtual device could not be found. Check to see if there is an MSS mount in process for the required system volume. If so, build a CPEVBLOK for this request, put the address in the MSSTASK3 field of the MSSCOM, and exit to DMKDSP.

| PROCESS A STAGING ADAPTER CYLINDER FAULT

| DMKIOSIN
| Entry when ending status is received from a device. Check to see if the CSW contains CE-DE with no error status.

| DMKIOS, TESTCYL
| If the device type is a 3330V, see if the CE-DE is in the MSS prefix NOP CCW. If not, or if the device is dedicated as a virtual 3330V, stack the IOBLOK.

| DMKSSS12
| Set the IOBFLT flag, indicating that a cylinder fault is being resolved. Chain the IOBLOK from the REDEVFIOB field in the RDEVBLOK. Build a TRQBLOK to recognize missing attentiCn interruptions and put it on the timer queue. Exit to the dispatcher.

| PROCESS AN ATTENTION INTERRUPT FROM A 3330V

| DMKIOS, IOSUNSOL
| Entry to process unsolicited I/O interrupts.

| DMKIOS, CALLMSSA
| If the interrupt is an attention, the device is a 3330V, and it is not dedicated as a 3330V. Call DMKSSSI1 to restart I/O.

| DMKSSSI1
| Find each IOBLOK for this device that has the IOBFLT flag set. Find the associated timer queue element and remove it from the timer queue. Turn off IOBFLT so that the IOBLOK can be restarted when the device is available.

READER'S
COMMENT
FORM

Title: IBM Virtual Machine Facility/370:
System Logic and Problem Determination
Guide Volume 1

Order No. SY20-0886-1

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

Which of the following terms best describes your job?

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

How did you use this publication?

- ☐ Introductory text
- ☐ Reference manual
- ☐ Student/ ☐ Instructor text
- ☐ Other (explain) _____

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

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

Specific criticisms (explain below)

Clarifications on pages _____

Additions on pages _____

Deletions on pages _____

Errors on pages _____

Explanations and other comments:

Thank you for your cooperation. No postage necessary if mailed in the U.S.A.

Note: Staples can cause problems with automated mail sorting equipment.
Please use pressure sensitive or other gummed tape to seal this form.

Trim Along This Line

Reader's Comment Form

Cut or Fold Along Line

Fold and tape — Please Do Not Staple — Fold and tape

NO POSTAGE NECESSARY IF MAILED IN THE UNITED STATES

BUSINESS REPLY MAIL
FIRST CLASS PERMIT 40 ARMONK, NEW YORK

POSTAGE WILL BE PAID BY ADDRESSEE:

International Business Machines Corporation
Department D58, Building 706-2
PO Box 390
Poughkeepsie, New York 12602

Attn: VM/370 Publications

Fold and tape — Please Do Not Staple — Fold and tape

IBM

International Business Machines Corporation
Data Processing Division
1133 Westchester Avenue, White Plains, N.Y. 10604

IBM World Trade Americas/Far East Corporation
Town of Mount Pleasant, Route 9, North Tarrytown, N.Y., U.S.A. 10591

IBM World Trade Europe/Middle East/Africa Corporation
360 Hamilton Avenue, White Plains, N.Y., U.S.A. 10601

IBM Virtual Machine Facility/370 System Logic and Problem Determination Guide Vol 1 Printed in U.S.A. SY20-0886-1

IBM Virtual Machine Facility/370 System Logic and Problem Determination Guide Vol 1 Printed in U.S.A. SY20-0886-1

International Business Machines Corporation
Data Processing Division
1133 Westchester Avenue, White Plains, N.Y. 10604

IBM World Trade Americas/Far East Corporation
Town of Mount Pleasant, Route 9, North Tarrytown, N.Y., U.S.A. 10591

IBM World Trade Europe/Middle East/Africa Corporation
360 Hamilton Avenue, White Plains, N.Y., U.S.A. 10601