File No. S370-37
Order No. SY20-0882-4

Systems

IBM Virtual Machine Facility/370: Service Routines Program Logic

Release 6 PLC 1

This publication describes the program logic for the VM/370 service routines. Each service routine (or group of routines) is described in its own chapter. Each chapter contains an introduction, a method of operation section, a program organization section, a directory, a data areas section, and a diagnostic aids section, if the section is applicable.

The service routines that are described in this publication are: the IBCDASDI Virtual Disk Initialization Program, the Interactive Problem Control System (IPCS), the Format Service Program, the DMKDIR Directory Program, the Installation Verification Procedure, the Procedures for Generating and Updating VM/370, the VM/370 Starter System, the 3704/3705 Service Programs, the ZAP Service Program, the EREP/Error Recording Interface, the MSS Communicator, and the IEBIMAGE Interface.

Fifth Edition (March 1979)

This is a major revision of SY20-0882-3 and makes obsolete that edition and Technical Newsletters SN25-0402, dated August 31, 1977, SN25-0415, dated December 15, 1977, and SN25-0440, dated July 15, 1978. This edition corresponds to Release 6 PLC 1 (Program Level Change) of IBM Virtual Machine Facility/370, and to all subsequent releases unless otherwise indicated in new editions or Technical Newsletters.

Extensive changes have been made to this publication; therefore, the user should read it in its entirety.

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

The "Introduction" describes the format of this publication, with special emphasis on using the method of operation diagrams.

The first chapter of this publication, "IBCDASDI--The Virtual Disk Initialization Program," describes the program that initializes virtual disks and assigns alternate tracks to disks that are used by VM/370.

The "IPCS--Interactive Problem Control System" chapter describes the logic for the commands that track and report both CP and non-CP problems.

The program that formats disks so they can be used by VM/370 is described in the "Format Service Program" chapter.

The "DMKDIR--The Directory Program" chapter describes the program that creates the VM/370 directory.

The "DASD Dump Restore Program" chapter describes the program that dumps, restores, and copies system disk files.

The "Installation Verification Procedure" chapter describes the EXEC procedure that checks the accuracy of the starter or newly generated system.

The "Procedures for Generating and Updating VM/370" chapter describes the EXEC procedures and modules that apply updates to the system, load the system, and generate new macro libraries.

The "VM/370 Starter System" chapter describes the system that is distributed to be used for system generation.

The "3704/3705 Service Programs" chapter describes the programs that perform generation and service functions for the control program for the IBM 3704/3705 Communications Controllers.

The "ZAP Service Program" chapter describes the program that modifies and dumps MODULE, LOADLIB, and TXTLIB files.

The "DMSIFC and DMSREA--EREP/Error Recording Interface" chapter describes the modules that interface between CMS and the OS/VS EREP program.

The "DMKMSS -- The MSS Communicator" chapter describes the program that operates in a virtual machine under OS/VS and interfaces between VM/370 and the MSS Mass Storage Control.

The "DMKIMG and DMKNMT -- IEBIMAGE Interface" chapter describes the utility programs required to dynamically change the character arrangement tables, graphic modifications, copy modifications, and FCBs for the 3800 Printing Subsystem.

In this publication:

- The term "2305 series" is used in reference to the IBM 2305 Disk Storage, Models 1 and 2.
- The term "3330 series" is used in reference to the IBM 3330 Disk Storage, Models 1, 2, and 11, and the IBM 3333 Disk Storage and Control, Models 1 and 11; and the 3350 Direct Access Storage operating in 3330/3333 Model 1 or 3330/3333 Model 11 compatibility mode.
- The term "3340 series" is used in reference to the IBM 3340 Disk Storage, Models A2, B1 and B2, and the 3344 Direct Access Storage Model B2.
- The term "3350 series" is used in reference to the IBM 3350 Direct Access Storage Models A2 and B2 in native mode.

  The terms "3705," "370X," and "3704/3705" include the IBM 3704, 3705-I, and 3705-II Communications Controllers, unless otherwise specified.
- Any mention of the IBM 2741 Communication Terminal also applies to the IBM 3767 Communication Terminal, unless otherwise stated.
- Information on the IBM 3705-II Communications Controller and the Type 4 Channel Adapter is for planning purposes only until the availability of the product.

Refer to the following publications for related material:

IBM Virtual Machine Facility/370:

CMS Command and Macro Reference, Order No. GC20-1818

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

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

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

Operatcr's Guide, Order No. GC20-1806

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

System Logic and Problem Determination Guide, Order No. SY20-0885

System Messages, Order No. GC20-1808

Interactive Problem Control System (IPCS) User's Guide, Order No. GC20-1823

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

OS/VS1 Mass Storage System Communicator (MSSC) Logic, Order Nc. SY35-0012

OS/VS2 Mass Storage System Communicator (MSSC) Logic, Order Nc. SY35-0013

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

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

OS/VS1 Programmer's Reference Digest, Order No. GC24-5091

OS/VS2 System Debugging Library: Debugging Handbcck, Order Nc. GC28-0632

IBM 3800 Printing Subsystem Programmer's Guide (CS/VS1, OS/VS2), Order No. GC26-3846

Concepts of the IBM 3800 Printing Subsystem, Order No. GC20-1775

Reference Manual fcr the IBM 3800 Printing Subsystem, Order Nc. GA26-1635

Contents

ILLUSTRATIONS
FIGURES

DIAGRAMS

MSS COMMUNICATOR

New: Program and Documentation

A new chapter "DMKMSS--The MSS Communicator" has been added that describes the program that operates in a virtual machine and interfaces between VM/370 and the MSS Mass Storage Control.

3800 TABLE CONSTRUCTION

New: Program and Documentation

A new chapter "DMKIMG and DMKNMT--IEBIMAGE Interface" has been added that describes the utility programs that can be used to change the character arrangement tables, graphic modifications, copy modifications and FCBs available on the 3800 Printing Subsystem, construct a directory and load it into a named system.

MISCELLANEOUS

The Chapter contents have been deleted from their original position at the beginning of each Chapter and now appear as a single Table of Contents at the beginning of the publication.

Summary of Amendments
for SY20-0882-3
as updated by SN25-0440
VM/370 Release 5 PLC 6

3340/3344 ALTERNATE TRACK SUPPORT

New: Error Recovery Support

Software error recovery procedures in CP 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. Similar recovery procedures have also been added to three stand-alone utilities: the DASD Dump Restore (DDR) Service Program; Directory Service Program (DMKDIR); and the Format/Allocate Service Program (DMKFMT). This will enable these programs to deal with 3340/3344 disks having flagged tracks. The changes to this publication reflect the changes made to the three utility programs.

CPEREP SUPPORT OF 3031, 3032, AND 3033 PROCESSORS

New: Hardware and Program Feature

VM/370 now supports the IBM 3031, 3032, and 3033 processors which have logout formats different from those of the other System/370 processors. During the initialization of the MCH/CCH error recording cylinders, frames are read from the Service Record File (SRF) device and written to the cylinder as a new record type. During CPEREP processing, both MCH and CCH records are formatted via the information contained in the frames on the cylinder.

INTERACTIVE PROBLEM CONTROL SYSTEM AND VMFDUMP

Changed: Program Feature and Documentation

The Interactive Problem Control System (IPCS) is now the VM/370 problem determination vehicle; therefore, the VMFDUMP (CP Abend Dump) Program chapter has been deleted and replaced with IPCS.

Introduction

This publication explains the program logic for each of the VM/370 service routines. Because the service routines are unrelated, they are discussed separately. One chapter of this publication is dedicated to each service routine (or logical group of service routines).

Each chapter is structured similarly. The following sections, where they are applicable, are included in each chapter:

- Introducticn
- Method of Operation
- Program Organization
- Directory
- Data Areas
- Diagnostic Aids

The first section, the "Introduction," gives a brief description of the service routine. This secticn explains what functions the service routine performs and tells how the program can be executed.

The second section, "Method of Operation," describes the program logic for the service routine. Diagrams describe the functions that the service routine performs and the "Notes" section of each diagram relates the function performed to the coding in the program. The labels of the related program sections are identified so that you can easily find the area in the program listing.

The "Program Organization" section contains a variety of information, such as entry points, data areas, and register usage. If the service routine is complex, there is a synopsis of the program modules or program routines.

The "Directory" lists all the program labels that are mentioned in the method of operation diagrams with a cross reference list indicating the diagram on which they appear. Also, there is a brief description of the function performed at the point in the program corresponding to each label. If the service routine contains more than one module, the correct module is indicated. The "Directory" is intended to help you quickly locate the section of the chapter that describes a particular function.

The "Data Areas" section contains detailed descriptions of the control blocks and data areas used by the service routine.

The last section, "Diagnostic Aids," contains a cross-reference list of the messages issued by the service routine. The message number and text are included with a label in the program reasonably close to the point where the message is issued. Messages are usually helpful when debugging a program problem.

Illustrations

There are two types of illustrations in this publication:

- Figures
- Diagrams

FIGURES

All general illustraticns, such as data areas and relationship drawings, are called "Figures". Figures may appear in any section of this publication.

DIAGRAMS

The method of operation drawings are called "Diagrams". Diagrams consist of a drawing and, very often, complementary notes. The drawing has three distinct parts:

- Process
- Input
- Output

The process block describes the action taken by the service rcutine. The input block shows the necessary input, such as data areas and control statements. The output block shows the resulting output, such as initialized disks cr copied files. The process block is found in the center of the drawing with the input block on the left and the output block cn the right. The Notes section appears below the drawing; it consists of a detailed comment, the module name (if the service rcutine consists of more than one module), the related program label, and a reference to any additional information (where appropriate).

Each step in the process block has a numbered key (1, 2, 3,...) and each substep has an alphabetic key (A, B, C,...). The related comment in the Notes section has the same key. The key that relates the processing step to a note is inside a box, and the key that relates a processing substep to a note is indented so that it is easily visible.

Illustration Numbering

Figures and diagrams are separately numbered. The format of the numbering system is:

```
Figure  X-nn
Diagram X-nn
```

where X designates the chapter (one through ten) and nn designates the relative position of the figure or diagram within the chapter. For example,

```
Figure 2-3
```

is the third figure in Chapter 2.

```
Diagram 3-1
```

is the first method of operation diagram in Chapter 3.

This publication is intended to acquaint the system programmer, and those programmers responsible for updating VM/370 service routines, with the operation of these service routines.

Chapter 1. IBCDASDI—The Virtual Disk Initialization Program

Introduction

The IBCDASDI program initializes virtual disks. A virtual disk is a logical division of a physical DASD that is used by OS and DOS for user direct access storage space with VM/370.

For virtual disk initialization, the following functions are performed:

1. Alternate tracks are assigned for those previously designated as defective.
2. A home address and track descriptor record are written.
3. IPL records are written on track 0 of the virtual disk.
4. A volume label is written on track 0 of the virtual disk.
5. A volume table of contents (VTOC) is constructed and written on the virtual disk.
6. An IPL program is written on track 0 of the virtual disk (if requested).
7. Surface analysis of each track is performed (if requested.) Alternate tracks are automatically assigned, if necessary.

For alternate track assignment, the following functions are performed:

1. The first available operative alternate track is selected from those indicated in the VTOC.
2. The address (CCHHR) of the primary track is written in the count field of the selected alternate track. The address (CCHHR) of the alternate track is written in the count field of the primary track.
3. The VTOC DSCB is modified to reflect the new status of available tracks.

<u>Note</u>: The IBCDASDI program does not assign alternate tracks or perform surface analysis for 3330 or 3350 disks.

INVOKING THE VIRTUAL DISK INITIALIZATICN PRCGRAM

The IECDASDI program may be loaded as a card deck, or as card images on tape. Control statements for the program can follow the last card (or last card image) for the program, or can be entered via a separate input device.

To execute the IBCDASDI prcgram:

1. Put a copy of the IECDASDI object deck into your virtual card reader, or mount and attach the tape containing the object program.
2. Load the object prcgram from the virtual reader on tape by issuing the CP IPL command for the appropriate virtual device address. When the program is loaded, an ebabled wait state is entered with the address field of the PSW containing the hexadecimal value FFFF.
3. When the program is lcaded and waiting for input, signal attention from the virtual console device. The message

   ```
   DEFINE INPUT DEVICE
   ```

   is sent to the virtual ccnsole. Enter the following response from the virtual console:

   ```
   INPUT=type cuu
   ```

   <u>where</u>:

   type is the virtual device type of the device containing the control statements. Valid device types are 1402, 1442, 2400, 2501, 2520, 2540, 3410, 3420, and 3505.

   cuu is the virtual device address cf the device containing the control statements.

Control statements are printed on the message output device. At the end of job, the END OF JOB message is printed cn the message output device and the program enters the wait state.

The IBCDASDI program initializes virtual disks whenever the CYLNO=nnn operand of the DADEF control statement is specified. The CYLNO=nnn operand specifies the number of cylinders to be initialized for a virtual disk. For information on running the IBCDASDI program and for a description of the necessary control cards see the *VM/370 Operator's Guide*.

Method of Operation

This section describes the execution of the virtual disk initialization (IBCDASDI) program and shows the processing associated with:

- Initializing virtual disks
- Assigning alternate tracks.

Figure 1-2 shows the relationship of the diagrams.

Diagram 1-1 describes the major functions of the virtual disk initialization program.

Diagram 1-2 shows how tracks are initialized.

Diagram 1-3 shows how alternate tracks are assigned.

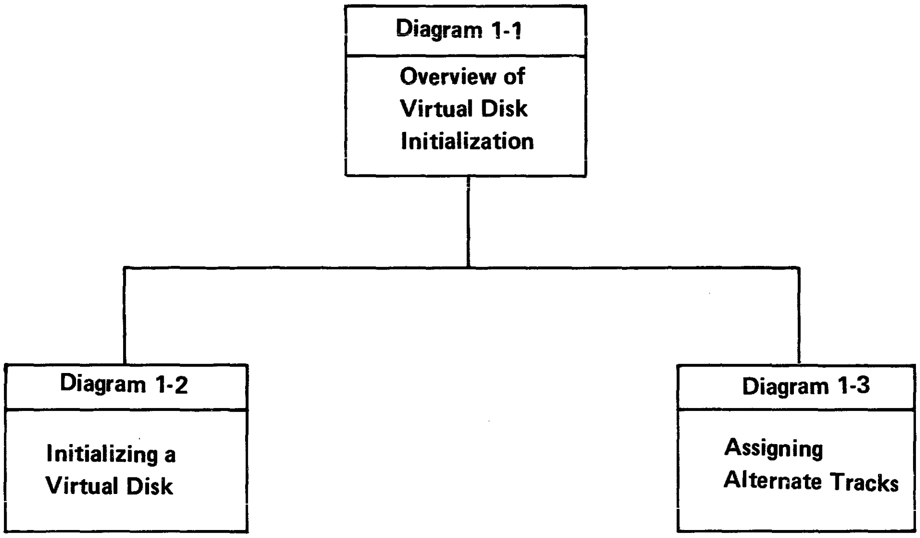
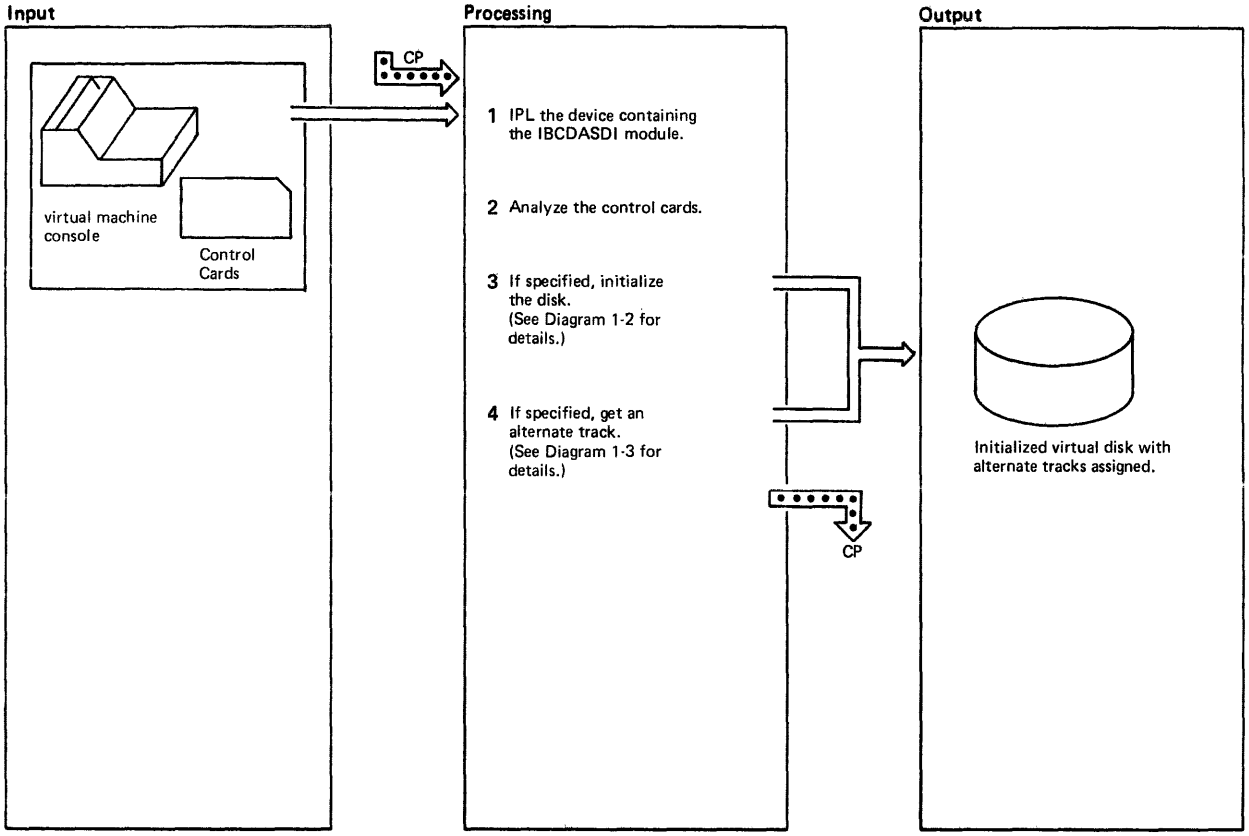


Diagram 1-1
Overview of Virtual Disk Initialization
Diagram 1-2
Initializing a Virtual Disk
Diagram 1-3
Assigning Alternate Tracks

Figure 1-1. Key to Virtual Disk Initialization Method of Operation Diagrams



1 IPL the device containing the IBCDASDI module.

2 Analyze the control cards.

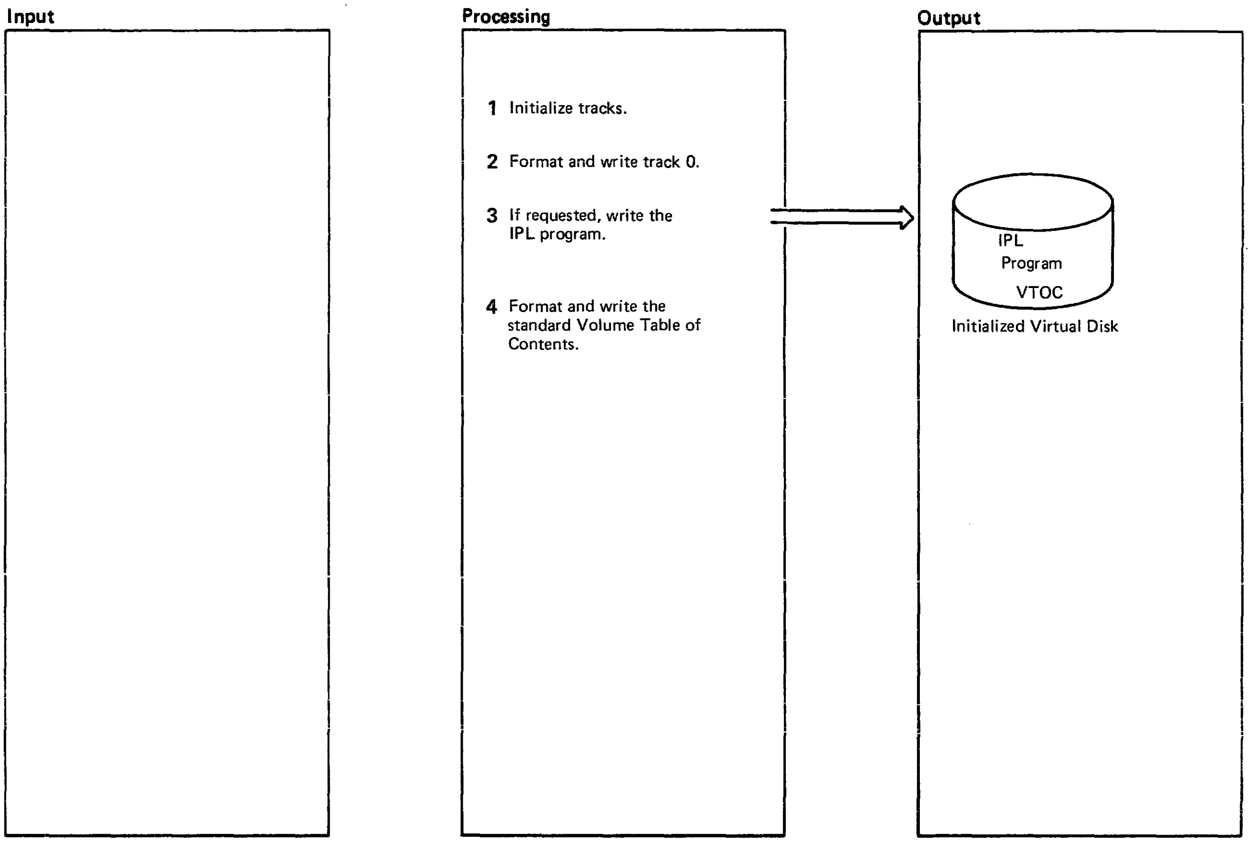
3 If specified, initialize the disk. (See Diagram 1-2 for details.)

4 If specified, get an alternate track. (See Diagram 1-3 for details.)

Notes	Module	Label	Ref
1 The general registers are cleared and the PSWs are initialized. The SEREP flags in the machine check old PSW are set to X'FF' and the program enters a wait state with X'FFFF' in the address field of the PSW.	IBCDASDI	START	
An attention interrupt identifies the virtual machine console and the input statement is read. The control cards may be read from a 1402, 1442, 2400, 2501, 2520, 2540, 3410, 3420, or 3505 input device.		CKINPUT	
2 Next, the CLRSCAN routine links to the control statements scan routine at RDCARD. RDCARD returns a pointer to a field and the length of the field in registers SCANADR and LENGTH, respectively, and an indication of the field type in location SWITCHRD. SWITCHRD is a one-byte switch with the following settings: Value Meaning X'50' Control statement error X'40' Bypass X'10' First control statement has been read X'08' Operator found X'04' Keyword found X'02' Parameter found	IBCDASDI	CLRSCAN	

Notes	Module	Label	Ref
3 The initialization routine is entered at VOLCHK if the volume label is to be checked and at GEN5E if the volume label is not to be checked. Initialization starts with the INTALT routine.	IBCDASDI	VOLCHK GEN5E	
4 Control is passed to the GETALTYZ routine if an alternate track is to be assigned. If volume label checking is requested, the CKVOLLBL routine does the checking. Assignment of alternate tracks starts with the GETALTX routine.	IBCDASDI	GETALTYZ	

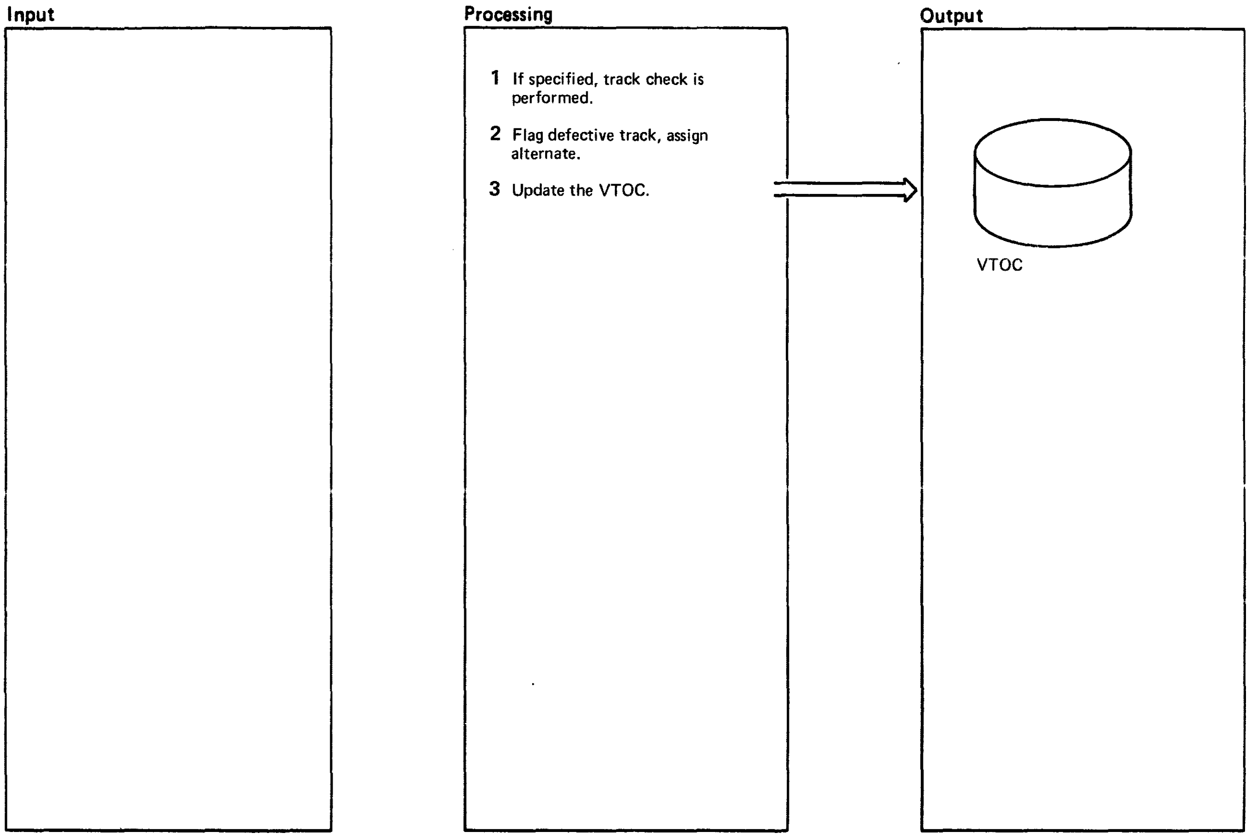
Diagram 1-1. Overview of Virtual Disk Initialization


Input
Processing
Output
1 Initialize tracks.
2 Format and write track 0.
3 If requested, write the IPL program.
4 Format and write the standard Volume Table of Contents.
IPL
Program
VTOC
Initialized Virtual Disk

Notes	Module	Label	Ref
1 When the defective flag test is suppressed, the home address is written followed by a maximum length record 0 consisting of hexadecimal 55. The track is read and checked. A maximum-length record 0 is written again, this time consisting of hexadecimal 00. The track is again read and checked. If a data check is encountered, this write sequence is repeated ten times. If data errors still occur, the track is flagged as defective. An alternate track is assigned when the device is disk. For drum devices, a message is issued indicating the address of the defective track. IBC101I BAD TRACK cccchhhh If the home address-record 0 area is defective on a 2314 or 2319 disk storage volume, an attempt is made to move the home address-record 0 fields down the track approximately 800 bytes.	IBCDASDI	INTALT	
2 Track 0 is written. It consists of two IPL records (or a dummy IPL record), a standard volume label and up to seven additional labels.	IBCDASDI	CONSTR2	

Notes	Module	Label	Ref
3 The user-supplied IPL initialization program is written. The program is written on cylinder 0 track 0 or 1, or, if the track is defective, on its assigned alternate.	IBCDASDI	WRTIPL	
4 The DSCBs needed for the VTOC are constructed: the DSCB (format 4) and the DADSM DSCB (format 5). For the 3330 series, the alternate track field in the format 4 DSCB is set to zero since no alternate track cylinder is provided. Then, the VTOC is written at the user-specified location and normal end of job is initiated. Control returns to CP.	IBCDASDI	FMTVTOC WRTVTOC	

Diagram 1-2. Initializing a Virtual Disk



1 If specified, track check is performed.

2 Flag defective track, assign alternate.

3 Update the VTOC.

Notes	Module	Label	Ref
1 If the track bypass is not selected, a track check is performed on the user-specified track. If the track is good, a message is issued. IBC109I TRACK CHK INDICATES TRACK IS GOOD	IBCDASDI	GETALTX GETALT2	
2 The ASGNALT routine flags the given track as defective and assigns an alternate. If the defective track is a primary track, the primary track is flagged and an alternate track is assigned. If the defective track is an alternate track that was not assigned to a primary track, the alternate track is flagged as defective. However, if the defective track is an alternate track that was assigned to a primary track, the alternate track is flagged and another alternate is assigned to the primary.	IBCDASDI	ASGNALT	
Then, the TRKPRNT routine causes a message to be printed stating the address of the defective track and its assigned alternate. IBC110I BAD TRACK cccchhhh IBC111I ALTERNATE cccchhhh		TRKPRNT	
3 Field six of the VTOC is decremented to indicate that one less alternate track is available, field five is incremented to point to the next available alternate track.	IBCDASDI	GETALT4	
Control is then given to location GETALT to repeat the process for the next user-specified track, or, if none exists, to initiate normal end of job. Control returns to CP.			

Diagram 1-3. Assigning Alternate Tracks

Program Organization

IBCDASDI -- GENERAL INFORMATION

VM/370 initializes virtual disks with the OS utility program IBCDASDI. IBCDASDI formats real or virtual VM/370 disk volumes for OS, DOS, and VSAM use. It should not be used to format CP disk areas (for paging, spooling, and so forth), or non-VSAM CMS disk areas. The execution of IBCDASDI is performed from the virtual card reader.

Initializing a Minidisk

IBCDASDI can, in addition to initializing real disks, initialize a minidisk. A minidisk can be initialized with or without a surface analysis (a test for defective tracks); a surface analysis should be included when a minidisk is initialized for the first time. Tracks in the last cylinders of a 2314 minidisk are left open for assignment as alternate tracks. No tracks are saved for alternates on 3330, 3340, or 3350 minidisks.

PROGRAM DESCRIPTION

This section describes the program logic of the IBCDASDI program.

Initialization with Surface Analysis

The IBCDASDI program does the following:

- For non-3330 devices:
  - -- Checks for tracks that were previously designated as defective (flagged) and have had alternates assigned.
  - -- Automatically assigns alternate tracks for 2314/2319 disk devices but not 3330s, 3340, and 3350.

  Note: This test must be suppressed when a disk is being initialized with surface analysis for the first time. This test must not be suppressed when a disk is initialized without surface analysis.

- For 2314 and 2319 devices:
  - -- Performs a surface analysis of each track
  - -- Automatically assigns alternates, if necessary.
  - -- Tracks that are available for use as alternates are checked first.

- For all devices:
  - -- Writes a track descriptor record (record 0), and erases the remainder of each track. IBCDASDI also writes a standard home address.
  - -- Writes IPL records on track 0 (records 1 and 2).
  - -- Writes volume label on track 0 (record 3) and provides space for additional records, if requested.
  - -- Constructs and writes a volume table of contents (VTOC).
  - -- Writes IPL program, if requested, on track 0 (for all DASD devices except 2302, 2303, and 2311) or track 1 (for 2302, 2303, and 2311).

Initialization without Surface Analysis

- For all devices except 3330 and 3340, checks for tracks that were previously designated as defective (flagged) and have had alternates assigned. The program automatically assigns alternates (2314/2319 disk devices only). This test must not be suppressed.

- For 3340 and 3350 devices, rewrites the home address and track descriptor record on all tracks. Tracks flagged defective are surface analyzed and reclaimed if no errors are detected. The BYPASS and FLAGTEST options are ignored. The PASSES=0 option will cause "QUICK DASDI" to be performed.

- For all devices:
  - -- Writes a standard home address, a track descriptor record (record 0), and erases the remainder of each track.
  - -- Writes IPL records on track 0 (records 1 and 2).
  - -- Writes volume label on track 0 (record 3) and provides space for additional records, if requested.
  - -- Constructs and writes a volume table of contents (VTOC).
  - -- Writes IPL program, if requested, on track 0 (for all DASD devices except 2302, 2303, and 2311) or track 1 (for 2302, 2303, and 2311).

Note: The IBCDASDI program can assign alternate tracks for real 3330/3340/3350 volumes only when they are specified by the GETALT statement. Even with the GETALT statement, the IBCDASDI program cannot assign alternate tracks for a 3330/3340/3350 minidisk because no cylinder has been allocated on which to assign alternate tracks. Defective tracks are flagged and alternate tracks are assigned when the 3330/3340/3350 storage volumes are initialized at the factory. An IBCDASDI job that initializes a 3330/3340/3350 performs the "Quick DASDI" function, which reads alternate tracks, decrementing the total unit of alternates by one whenever an alternate is found defective or assigned, writes a volume label and VTOC, and writes an IPLTEXT if requested. No surface analysis is performed and no home address or record 0 is written on the primary tracks. The BYPASS and FLAGTEST options of the DADEF statement are ignored.

DASD 3340 disk packs are factory-shipped without flagged tracks and alternate track assignments. IBCDASDI's "Quick DASDI" detects 3340 customer-generated alternate track assignments.

The IBCDASDI program cannot check to see if the 3330, 3340, or 3350 space to be initialized was previously formatted.

Entry Point
START

The absolute loader gives control to the START routine. After the control statements are read and analyzed, control is passed to:

VOLCHK if volume label checking and virtual disk initialization are requested.

GEN5E if virtual disk initialization without volume label checking is requested.

GETALTYZ if alternate track assignment is requested.

Data Areas, Tables and Workareas

OPRTAB A list of valid command words used on control cards.

KEYTAB A list of valid key words used on control cards.

SYINTAB A list of valid input devices for control cards.

DEVTAB A list of valid devices handled by the IBCDASDI program.

PROCBUFF A list of constants for specific device types.

CONVTAB A table used to convert console input to upper case.

IOTAB A table used to convert hexadecimal to EBCDIC graphic.

ALTTT A table used to convert EBCDIC graphic to hexadecimal.

TAB13 A buffer of hexadecimal zeros (X'00') for disk and drum surface analysis.

TAB5F A buffer of hexadecimal E5s (X'E5') for data cell surface analysis. Also used as a buffer of hexadecimal 55s (X'55') for disk and drum surface analysis.

Routines Called
None

External References
The program enters an enabled wait state shortly after it is loaded. An attention interrupt identifies the virtual machine console and the INPUT control statement is read. This control statement tells IBCDASDI which device to read to get the program control statements.

Exit Conditions
Successful completion of DMKMDA results in a wait PSW with E's in the address field. An end of job message is also given on the message device.

Error exits result in E's being loaded in the address field of a wait PSW. Error stops, whenever possible, are preceded by a message which defines the error. Machine check interrupts result in the loading of a PSW in the wait state with E2 in the address field.

Directory

Figure 1-2 is an alphabetical list of some of the labels in the IBCDASDI module. The function performed at the point in the program indicated by each label is described and the associated method of operation diagram is referenced.

Label	Diagram	Description
ABCFOUND	1-1	Finds size of alternate track area.
ASGNALT	1-3	Assigns alternate tracks for defective tracks.
CKINPUT	1-1	Reads the control cards.
CLRSCAN	1-1	Scans control cards.
CONSTR2	1-2	Writes track 0.
DASDI	1-1	Performs program initialization and reads control cards.
FMTVTOC	1-2	Formats the VTOC.
GEN5E	1-1	Starts initializing virtual disks when volume label checking is not desired.
GETALTX	1-3	Gets an alternate track for a defective track.
GETALTYZ	1-1	Checks whether volume label checking was requested when assigning alternate tracks.
GETALT2	1-3	Checks to see if flagged track is really defective before assigning alternate. This check is optional.
GETALT4	1-3	Updates the VTOC after an alternate track assignment.
INTALT	1-2	Tests tracks during initialization.
START	1-1	Performs program initialization.
TRKPRNT	1-3	Issues message identifying defective track and the alternate assigned.
VOLCHK	1-1	Checks volume label when initializing virtual disks.
WRTIPL	1-2	Writes the IPL program on disk.
WRTVTOC	1-2	Writes the VTOC.

Figure 1-2. Virtual Disk Initialization Program Label Directory

Data Areas

This section describes the data areas used by the IBCDASDI Virtual Disk Initialization Program. The data areas are:

- Track Zero
- Unit Control Block

TRACK ZERO

```
┌────────────────────────────────────────────────────────────────>   >──────────────────┐
|  HA   |    RO     |     R1    |   R2    |  R3   |    R4      |<   <   |   R10         |
└────────────────────────────────────────────────────────────────>   >──────────────────┘
└───V───┴─────V─────┴─────V─────┴────V────┴───V───┴─────V──────┘         └──────V───────┘
 Home    Track       IPL Record IPL       Standard Additional             Additional
 Address Description or Dummy   Bootstrap Volume   Volume Label           Vclume Label
                                          Label    (Optional)             (Cptional)
```

Figure 1-3. IBCDASDI Track Zero

UNIT CONTROL BLOCK

The field UCBREG contains the address of the UCB. All references to fields in the UCB are made using offsets from UCBREG; the UCB fields are not labeled in the IBCDASDI program. The format of the UCB is:

```
┌──────────────────────────────────────────────────────────────────┐
│0              │1               │2                              │
│      *1       │    not used    │     Channel and unit number   │
│──────────────────────────────────────────────────────────────────│
│4              │5                                               │
│      *2       │               CAW                              │
│──────────────────────────────────────────────────────────────────│
│8                                                                 │
│                          Interrupt PSW                           │
│                                                                  │
│                                                                  │
│──────────────────────────────────────────────────────────────────│
│10                                                                │
│                          Interrupt CSW                           │
│                                                                  │
│                                                                  │
│──────────────────────────────────────────────────────────────────│
│18                                                                │
│                        Sense Information                         │
│                                                                  │
│                                                                  │
│──────────────────────────────────────────────────────────────────│
│20                                                                │
│                                                                  │
│                                                                  │
│           2305, 3330, 3340, 3350 Sense Information               │
│                                                                  │
│                                                                  │
│                                                                  │
│                                                                  │
└──────────────────────────────────────────────────────────────────┘
```

Displacement			
Hex	Dec	Field	Description
0	0	*1	Unit reference number
1	1		Not used
2	2		Channel and unit number
4	4	*2	CAW protect
5	5		CAW
8	8		Interrupt PSW
10	16		Interrupt CSW
18	24		Sense Information
20	32		2305, 3330, 3340, 3350 Sense Information

Figure 1-4. Unit Control Block

Diagnostic Aids

Figure 1-5 is a list of the messages issued by the Virtual Disk Initialization Program and the label of the routine that caused the message to be issued.

Message Code	Label	Diagram	Message Text
IBC101W	ERR235 ERR635		INVALID CARD CODE. CORRECT ERROR. DEPRESS INTERRUPT KEY.
IBC102A	SCANPRT		CONTROL STATEMENT ERROR. JOB TERMINATED.
IBC103A	ABSEQERR		STATEMENT SEQUENCE ERROR. JOB TERMINATED.
IBC104W	SVCINT		SVC INTERRUPT. JOB TERMINATED.
IBC105A	ATTN		DEFINE INPUT DEVICE.
IBC106A	CKVOLLBL		THE VOLID IN CONTROL STATEMENT DOES NOT AGREE WITH ID IN VOL MABEL WHICH FOLLOWS
IBC107W	TRKZERO		TRACK ZERO BAD. JOB TERMINATED.
IBC108A	TRACKBAD		HA OR RO FIELD BAD. JOB TERMINATED.
IBC108I	ALTE2314		HA OR RO FIELD BAD.
IBC109I	GETALT2	1-3	TRACK CHK INDICATES TRACK IS GOOD.
IBC110I	TRKPRNT	1-3	BAD TRACK cccchhhh.
IBC111I	TRKPRNT	1-3	ALTERNATE cccchhhh.
IBC112W	ALTERR		ALT TRACKS DEPLETED. JOB TERMINATED.
IBC113W	VTOCERR		IMPROPER VTOC BEGIN ADDRESS. JOB TERMINATED.
IBC153A	FAILREAD		TYPEWRITER FAILED TO READ LAST MESSAGE. DEPRESS INTERRUPT KEY.
IBC154A	ERR233 ERR633		READY READER cuu. DEPRESS INTERRUPT KEY.
IBC155A	ERR433 ERR833		READY PRINTER cuu. DEPRESS INTERRUPT KEY.
IBC156A	ERR333		READY TAPE cuu. DEPRESS INTERRUPT KEY.
IBC157A	ERR133		READY DASD cuu. DEPRESS INTERRUPT KEY.
IBC159A	ERR232 ERR632		READER CHECK. CORRECT ERROR. DEPRESS INTERRUPT KEY.
IBC160A	ERR432 ERR832		PRINT CHECK. CORRECT ERROR. DEPRESS INTERRUPT KEY.
IBC163A	EOJ		END OF JOB.
IBC201W	COMREJ		COMMAND REJECT
IBC202A	INTVREQ		INTERV. REQUIRED
IBC203W	BUSOUT		BUS OUT CHECK
IBC204W	EQUIPCHK		EQUIPMENT CHECK
IBC205W	DATACHK		DATA CHECK
IBC206W	OVRRUN		OVERRUN
IBC208W	CONVTERR		DATA CONV. CHECK
IBC209W	ENDOFCYL		END OF CYLINDER

Figure 1-5. IBCDASDI Messages (Part 1 of 2)

Message Code	Label	Diagram	Message Text
IBC210W	INVLDADR		INVALID ADDRESS
IBC211W	SIOERR		NOT AVAILABLE
IBC212W	LPDATA		READ DATA CHECK
IBC214W	TROVRRUN		TRACK OVERRUN
IBC215W	FILEPROT		FILE PROTECTED
IBC217W	NORECMSG		NO RECORD FOUND
IBC218W	ENDTEST		INVALID ERROR
IBC219W	NEVER		WRONG ERROR
IBC220W	CHANERR		CHAN. CTRL ERROR
IBC221W	INFCERR		INTERFACE ERROR
IBC222W	CHDATA		CHAN. DATA CHECK
IBC223W	OVRFLOW		DASD OVERFLOW
IBC224W	PROCHK		PROGRAM CHECK
IBC225W	PTNCHK		PROTECTION CHECK
IBC226W	UEERROR		UNIT EXCEPTION
IBC227W	WLRERR		INCORRECT LENGTH
IBC228W	CHAINCHK		CHAINING CHECK
IBC229W	SEQERR		COMMAND SEQ. ERR
IBC230W	SEEKCHK		SEEK CHECK ERROR
IBC231W	RITERROR		WRITE DATA CHECK
IBC232W	TAPELP		TAPE -- LOAD POINT
IBC234W	MISSMARK		MISSING ADR-MARK
IBC235W	BLNKSTOP		BLANK TRACK
IBC236W	BLNKSTOP		3 BLANK CYLINDER
IBC237W	BLNKSTOP		3 BLANK STRIPS
IBC239W	BLKTRCK1		3 BLANK TRACKS
IBC242W	INTRKFMT		INVALID TRK FMT
IBC243W	INHIBIT		WRITE INHIBITED
IBC249W	ERROROF		I/O ERROR. JOB TERMINATED.

Figure 1-5. IBCDASDI Messages (Part 2 of 2)

Chapter 2. IPCS—The Interactive Problem Control System

Introduction

The Interactive Problem Control System (IPCS) is a group of CMS commands which, when used in the VM/370 Control Program, track and report both CP and non-CP problems. The IPCS commands are:

> DUMPSCAN--which allows you to inspect CP dumps that the VMFDUMP command has converted to CMS files. It prompts you for the dump number and filemode, and it lets you enter subcommands to display specific parts of the dump and to locate data and addresses.
>
> PRB--which allows you to update the status, last update function, severity, and PTF (Program Temporary Fix) files of the symptom summary record for a problem.
>
> PROB--which allows you to describe a problem that is not a CP abend, or to add information to an existing problem report (whether or not it is a CP abend). It prompts you for all the necessary information about the problem.
>
> STAT--which allows you to produce a list of the status of all problems that you can print or type. You can also request the status of a single problem or a subset of problems and display it at the terminal.
>
> VMFDUMP--which allows you to convert CP dumps into CMS files, create problem reports, and search for duplicate problems.
>
> All that is necessary to use the IPCS commands is that the command modules be installed on your VM system and that the modules and IPCS files be available to the appropriate users.

IPCS REPORT FILES

Usually, all IPCS files reside on the A-disk of the user responsible for maintaining your VM/370 system. All files associated with a given problem (such as a dump or supplementary files) are of the form:

```
PRBnnnnn filetype
```

The number assigned to the problem by IPCS is indicated by nnnnn and the filetype is one of the following:

> DUMP--a CMS file; the output of the VMFDUMP command.
>
> REPORT--the report generated by the PROB command or the VMFDUMP command. (One exists for each problem known to the system.)

OTHER IPCS FILES

Other IPCS files include the NUC MAP file, the STATALL LOCAL file, the summary record, and the symptom summary.

NUC MAP is the nucleus load map of the CP dump being analyzed. It contains every module name and entry point in the CP nucleus and is required by the VMFDUMP command for successful analysis of the dump. An abbreviated version of the NUC MAP is appended to the VMFDUMP and is used by the DUMPSCAN command.

The STATALL LOCAL file contains the status of all problems known to the system and is created by the STAT command when entered with the ALL operand.

The summary record contains the next available problem number. It is a single 80-character record that is assigned to a problem when it is reported. The number is then increased by 1 and the summary record is rewritten.

The symptom summary contains the symptoms and status of each problem known to the system. There is one symptom summary control record for each problem that is created and placed in this file by the PROB and VMFDUMP commands. These records are displayed by the STAT command and updated by the PRB command. They are also used to identify possible duplicate problems as they are added to this file.

CP ABEND DUMPS

During system generation a user is designated to receive CP abend dumps. If an abend occurs when SET DUMP AUTO is in effect, and sufficient contiguous space is available in the CP paging area, the abend will appear in the designated user's virtual reader. The user can then use the VMFDUMP command to read the spool file, create a CMS file containing the dump, and print it.

After a CP dump is created, any user who has access to the IPCS commands and files can use them to examine the dump, the problem report, and the status of the problem. However, an IPCS file must be on the user's A-disk for him to update it.

Method of Operation

This section describes Interactive Problem Control System (IPCS). Diagrams describe the five IPCS functions. Figure 2-1 shows the relationship of these diagrams.

Diagram 2-1 shows how the DUMPSCAN command and its subcommands enable the user to interactively examine a CMS dump file created by VM/370.

Diagram 2-2 shows how the PRB command updates the status of problems in the symptom summary file.

Diagram 2-3 shows how the PROB command creates problem reports and adds information to existing problem reports.

Diagram 2-4 shows how the STAT command lists the current status of a given problem.

Diagram 2-5 shows an overview of how the VMFDUMP command creates a problem report by extracting pertinent data from a VM/370 CP abend dump.

Diagram 2-6 shows how the nucleus load map is compressed.

Diagram 2-7 shows how a program check is handled.

Diagram 2-8 shows how a coded abend is handled.

Diagram 2-9 shows how an operator initiated dump is handled.

Diagram 2-10 shows how the preliminary information is printed.

Diagram 2-11 shows how the control blocks are formatted and printed.

Diagram 2-12 shows how the storage protection keys and dump file are printed.

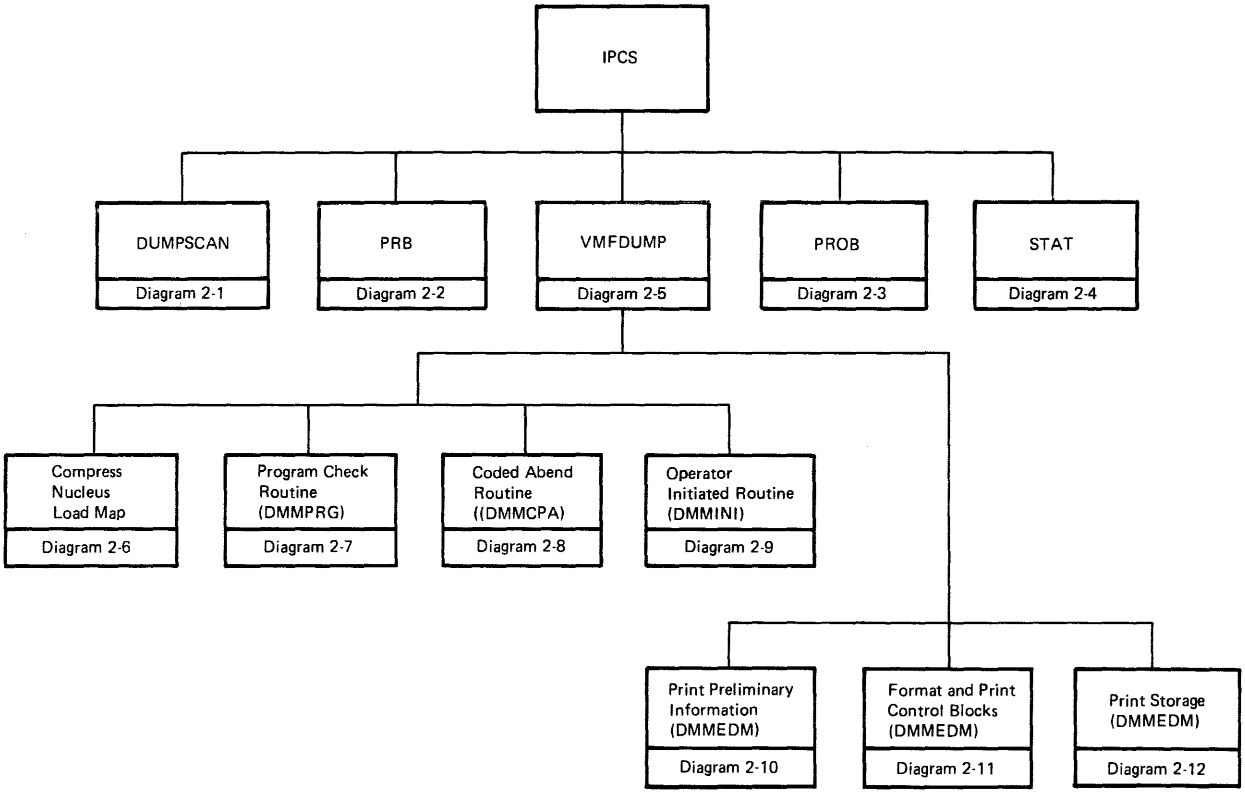
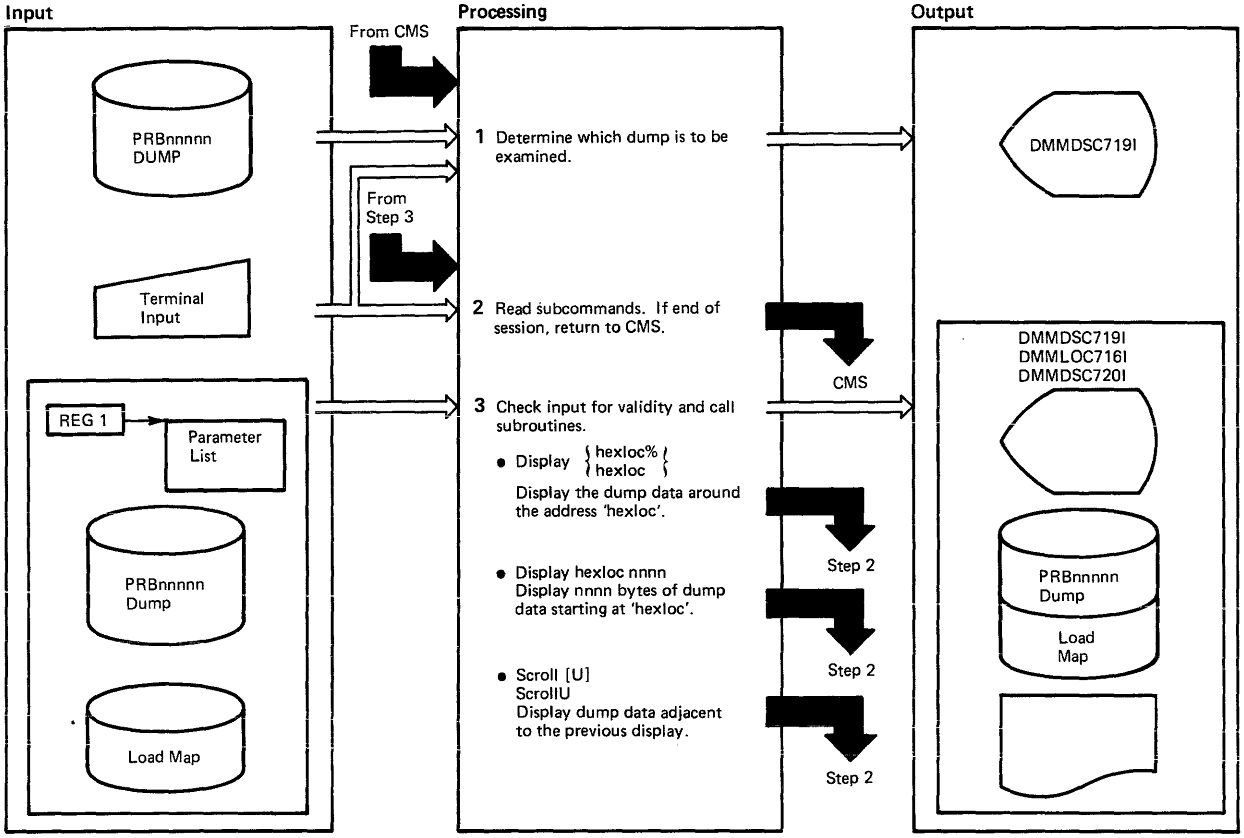

IPCS
DUMPSCAN
Diagram 2-1
PRB
Diagram 2-2
VMFDUMP
Diagram 2-5
PROB
Diagram 2-3
STAT
Diagram 2-4
Compress Nucleus Load Map
Diagram 2-6
Program Check Routine (DMMPRG)
Diagram 2-7
Coded Abend Routine ((DMMCPA)
Diagram 2-8
Operator Initiated Routine (DMMINI)
Diagram 2-9
Print Preliminary Information (DMMEDM)
Diagram 2-10
Format and Print Control Blocks (DMMEDM)
Diagram 2-11
Print Storage (DMMEDM)
Diagram 2-12

Figure 2-1. Key to Interactive Problem Control System Method of Operation Diagram



1 Determine which dump is to be examined.

2 Read subcommands. If end of session, return to CMS.

3 Check input for validity and call subroutines.

- Display { hexloc% / hexloc }
  Display the dump data around the address 'hexloc'.
- Display hexloc nnnn
  Display nnnn bytes of dump data starting at 'hexloc'.
- Scroll [U]
  ScrollU
  Display dump data adjacent to the previous display.

Notes	Module	Label	Ref
1 Ask if the user needs help. If yes, show 'HELP' pages. Prompt for the dump file number and mode. Check that the dump exists. If not, issue message:	DMMDSC	GETDUMP ENTER	
ERROR IN FSSTATE		STATERR	
2 Read subcommands from the terminal. Truncate the input to 8-byte words and place in parameter list. If end of session ('HX', 'QUIT', or 'END'), return to CMS.	DMMDSC	READY TOKEN INCHECK	
3 Scan the parameter list and check subcommand for validity. Pass control to the appropriate subroutine.	DMMDSC	PLISTSCN	
• Convert 'hexloc' to hexadecimal and call DMMGET to fetch the dump data into storage. Check for indirect request [%]. Call DMMDIR to format and display the data.	DMMFEX	TRANADD NOROUND INDIR DIRECT	
• Translate the address of 'hexloc' and the byte count nnnn to hexadecimal. Fetch the data into storage, fromat the data, convert to EBCDIC, add addresses, and write the data to the terminal.	DMMFED	DMMFED TRANADD CTRAN DIRECT SCRNFULL	
• Increase (decrease if 'U') the last displayed address by X'130', checking for an end of page condition. Fetch that data into storage. Invoke the format and display routine (DMMDIR).	DMMSCR	DMMSCR DIRECTIT	

Notes	Module	Label	Ref

Diagram 2-1. DUMPSCAN IPCS Command (Part 1 of 3)

Input

Processing

Output

3 Continued

- Trace $\begin{bmatrix}\underline{19}\\ \text{nnn}\end{bmatrix}$
  Display the lastest nnn entries in the trace table.

Step 2

- Locate string fromaddr toaddr [increment]
  Find the string in the dump and display it.

Step 2

- DMKmmee
  Use the load map to find the module in the dump and display the code.

Step 2

- Rioblok cuu
  Vioblok cuu
  Display the channel, control unit and device block.

Step 2

- Vmblok
  Display a list of logged on users and their status.

Step 2

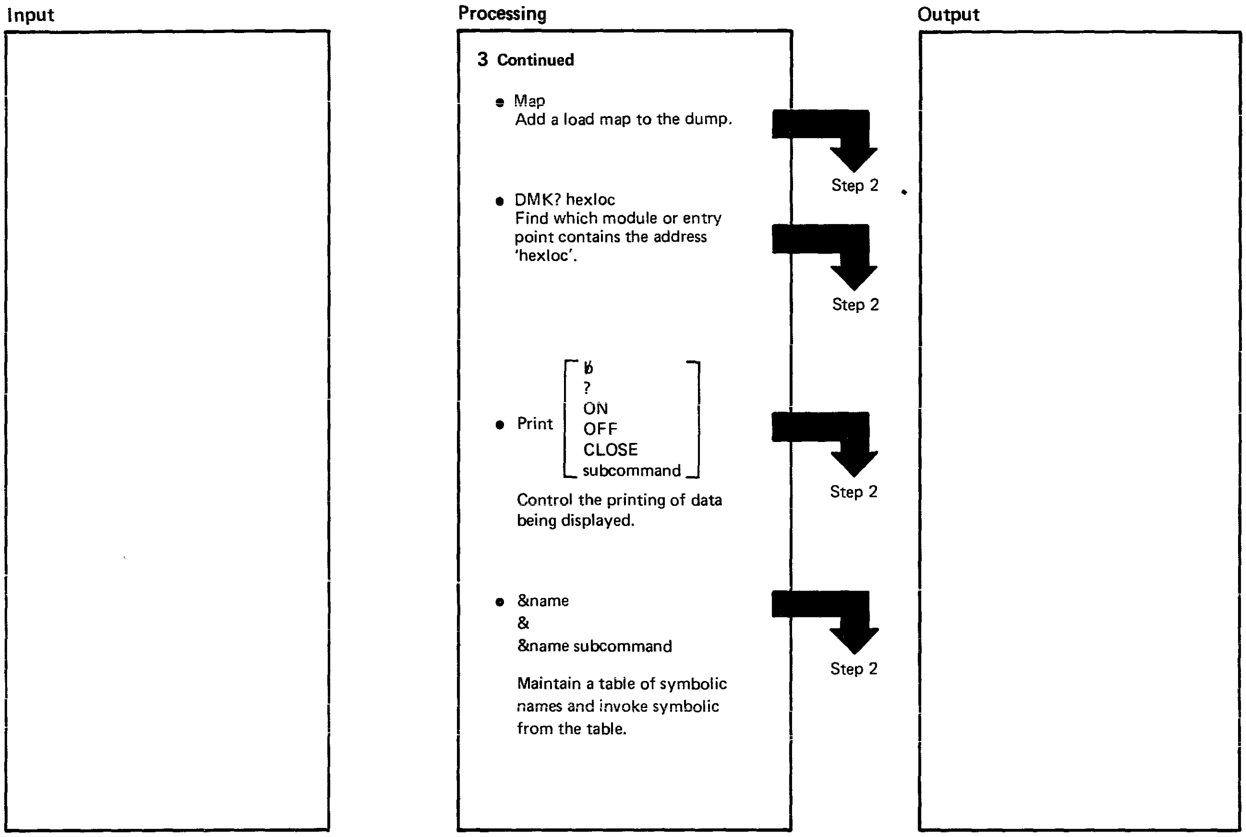
- $\begin{bmatrix}\text{A}\\ \text{M}\end{bmatrix}$REGS
  C
  G
  Display the registers and, optionally, the clocks, timers, PSWs, CSW, and CAW.

Step 2

Notes	Module	Label	Ref
3 Continued			
• Find the trace table from 'traccurr' in the PSA. Convert the count to a byte count. Submit it to DMMFED as display 'hexloc' nnnn.	DMMTRC	DMMTRC	
• Fetch the page containing the 'fromaddr' into storage. Compare the string against the data at 'fromaddr'. If not equal, increase the data pointer by the 'increment' parameter and compare again. Continue until either the comparison is equal then display the area containing the equal compare or until the 'toaddr' is reached then issue message:	DMMLOC	DMMLOC EXECUTOR GOGOFEX	
STRING 'string' NOT FOUND		MOREMSG	
• Read the load map from the end of the dump and scan it for this label. Submit the address of the label to DMMFEX to display.	DMMMOD	DMMMOD MAPRED	
• Separate the channel block, control unit block, and device block for the given real or virtual device address. Display the blocks.	DMMIOB DMMFED	DMMIOB COMPRCUB UIO DMMFED	
• Get the system VMBLOK pointer from the PSA. Follow the pointer to the chain of VMBLOKs. Print a list of the active VMBLOKs with the userid and selected status bytes.	DMMVMB	DMMVMB BALGET MOVEL	

Notes	Module	Label	Ref
• Check for AP or UP dump. Select the appropriate set of registers and if the subcommand is not 'C' or 'G', also display the PSW and clocks.	DMMREG	DMMREG	

Diagram 2-1. DUMPSCAN IPCS Command (Part 2 of 3)



Input

Processing

3 Continued

- Map
  Add a load map to the dump.
- DMK? hexloc
  Find which module or entry point contains the address 'hexloc'.
- Print [b̸ | ? | ON | OFF | CLOSE | subcommand]

  Control the printing of data being displayed.
- &name
  &
  &name subcommand

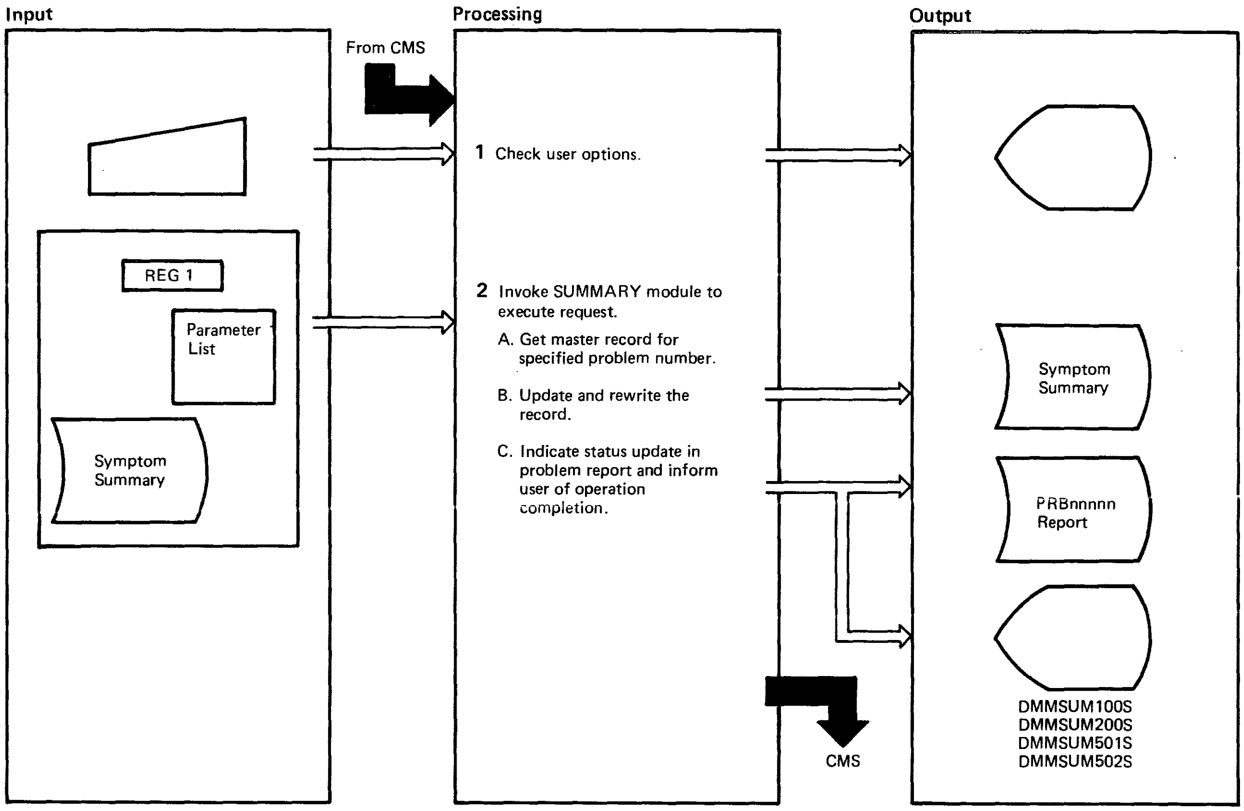
  Maintain a table of symbolic names and invoke symbolic from the table.

Output

Notes	Module	Label	Ref
3 Continued			
• Check that the dump does not already have a load map. If it does, issue the message:	DMMDSC	MAPCHECK	
LOAD MAP ALREADY PRESENT		TWOMAPS	
If it does not, call DMMMAP to add the load map to the dump. (See Diagram 2-6 for a description of DMMMAP processing.)			
• Read the load map from the end of the dump. Scan for the address closest to, and before the given address.	DMMMOD	READ QREQUEST	
See if the module is pageable. If it is, find its loaded address at dump time. Display the entry point name and displacement.		PAGEMOD	
• Turn PRINT 'ON' or 'OFF' as requested.	DMMDSC	CHECKTWO	
? Display the current print status.		SHOWPSW	
CLOSE Issue CP DIAGNOSE '08'.		CLOSEPRT	
subcommand Issue subcommand and turn printing 'ON' for subcommand.		SUBCOM	
PRINT Reissue the previous subcommand and print the output.		RESUBCOM	

Notes	Module	Label	Ref
• &name Call a names subcommand from the table.	DMMDSC	FOUNDAMP	
& Display a list of the entries in the table.		SHOWTAB	
&name subcommand Add the subcommand into a table of subcommands.		NOTINTAB	

Diagram 2-1. DUMPSCAN IPCS Command (Part 3 of 3)



1 Check user options.

2 Invoke SUMMARY module to execute request.

A. Get master record for specified problem number.

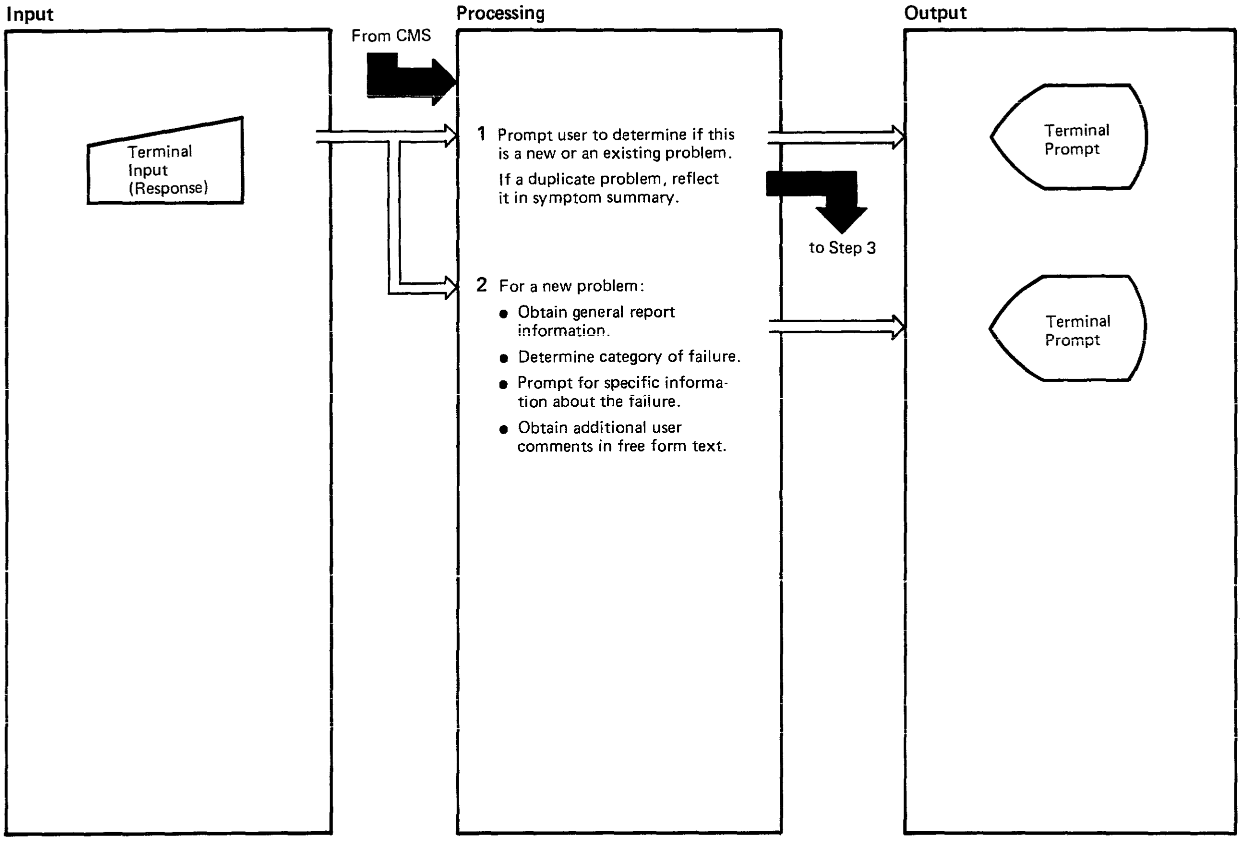
B. Update and rewrite the record.

C. Indicate status update in problem report and inform user of operation completion.

Notes	Module	Label	Ref
1 Check the first two operands (which may be in any order) as entered, swap them and check again. Issue error messages for invalid input.	PRB	–RETRY	
2 Call SUMMARY module (DMMSUM). The following routines supply information appropriate to the user's request: – PTFON – DSPLY – PTFIS – IBM – CLOSE – USER – DUPOF – NEEDINFO – APAR – HELP – SEV			
A. Pass input to parameter list pointed to by register 1. The parameter list is: **Byte Contents** 1-8 Not used 9-16 PRBnnnnn (problem number) 17-24 Request type: UPSTAT – Update status UPFUNCT – Update last function UPSEV – Update severity UPPTF – Update PTF information UPDUP – Update duplicate information UPAPAR – Update APAR information 25-32 Update specific information.	DMMSUM	START	

Notes	Module	Label	Ref
B. All activities cause "LAST" date to be updated with the current date.			
C. After the status is updated in the symptom summary file, append the date and time and new status to the problem report for history purposes. The SUMMARY module supplies a return code which is checked. If zero, an informational reply is issued indicating successful completion. If the completion code is not zero, an informational reply is issued indicating that the update was unsuccessful.	DMMSUM PRB	REPORT1	

Diagram 2-2. PRB IPCS Command



1 Prompt user to determine if this is a new or an existing problem.

If a duplicate problem, reflect it in symptom summary.

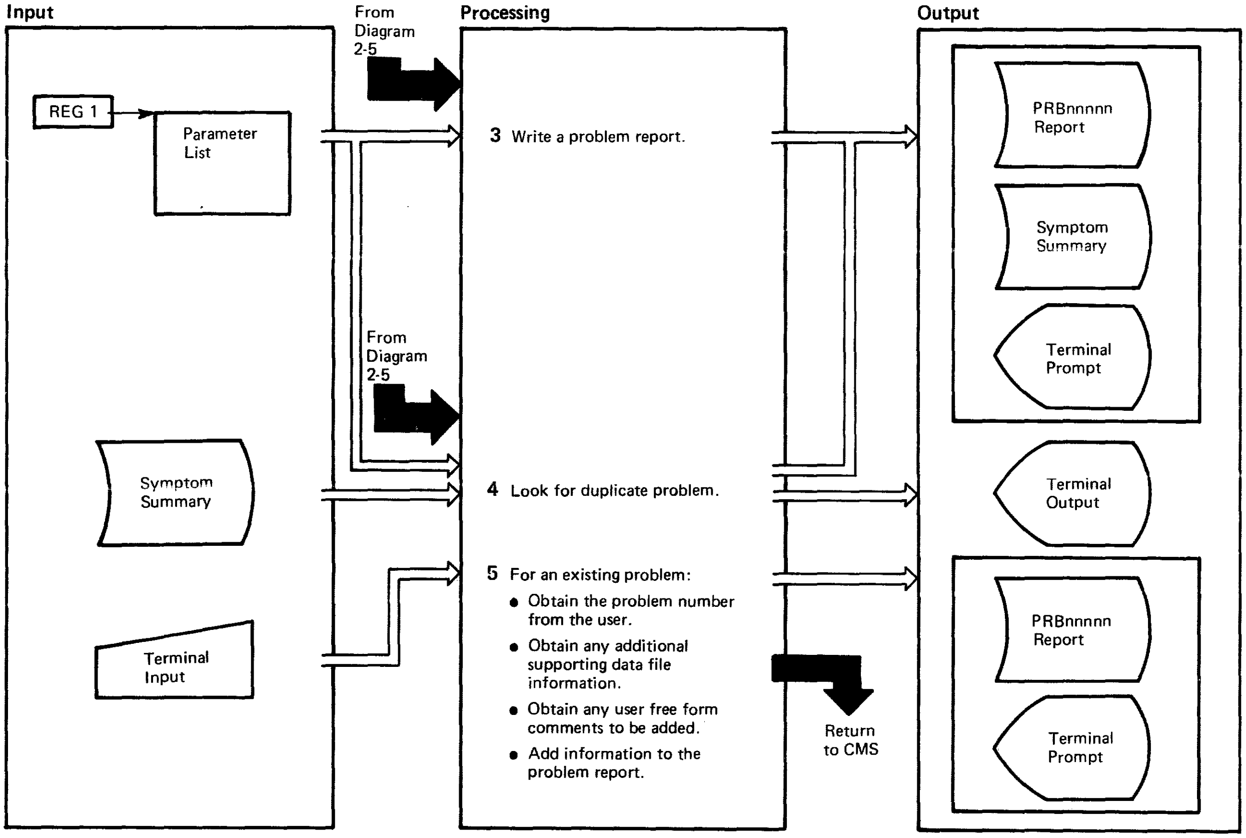
2 For a new problem:

- Obtain general report information.
- Determine category of failure.
- Prompt for specific information about the failure.
- Obtain additional user comments in free form text.

Notes	Module	Label	Ref
1 Prompt user to find out if this activity is to create a new problem report or to update an existing report.	DMMPRO	EXIST	
2 A response of 'NO' to this prompt indicates this is a new problem.	DMMPRO	MAINLINE	
● The user is prompted for the date and time of the failure, the SCP, CPU type, CPU serial, and other general information.			
● The user is prompted for the category of the problem; for example, abend message or loop.		GETFAIL	
● The user is prompted for detailed information, depending on the type of failure.			
● The user is prompted for additional user comments.		TEXTENTR	

Notes	Module	Label	Ref

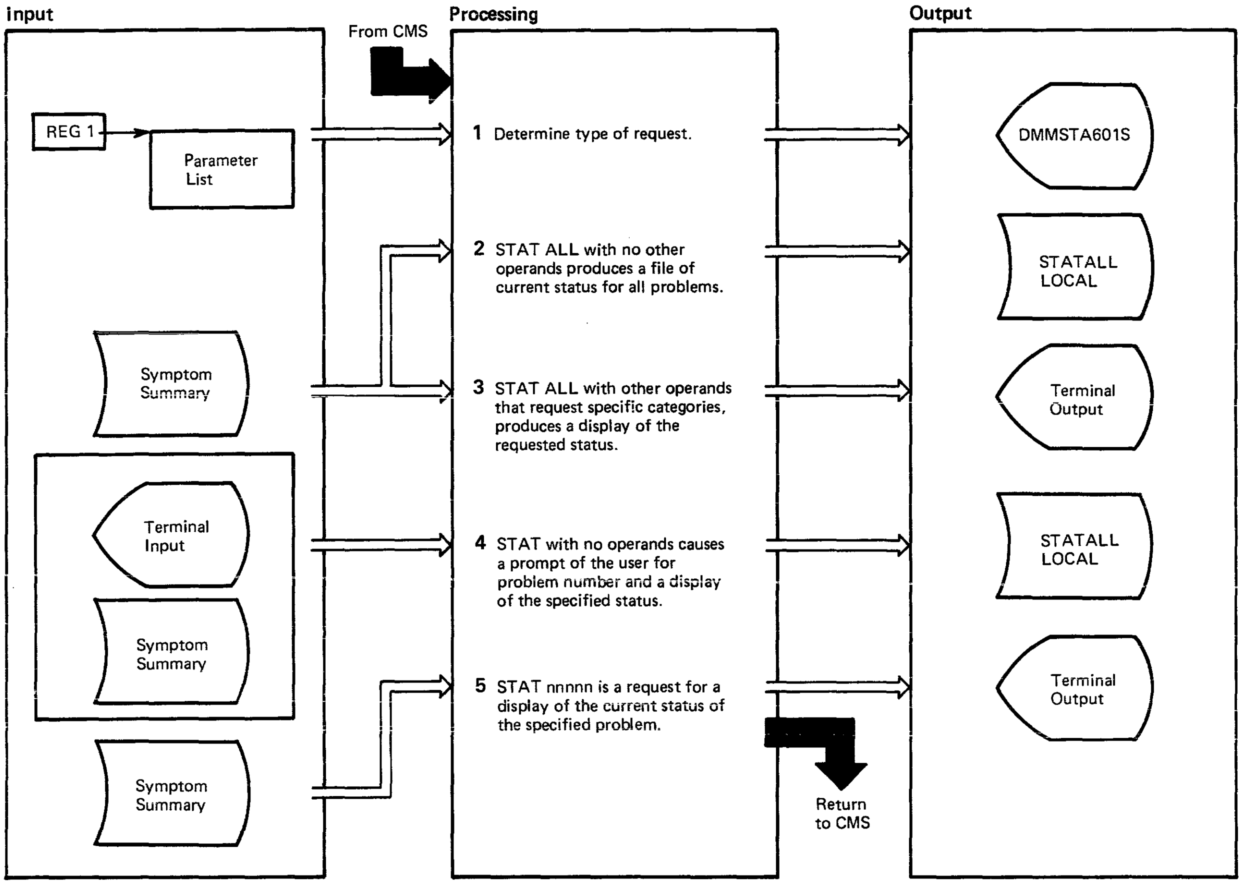
Diagram 2-3. PROB IPCS Command (Part 1 of 2)


Input
From Diagram 2-5
Processing
Output
REG 1
Parameter List
3 Write a problem report.
PRBnnnnn Report
Symptom Summary
Terminal Prompt
From Diagram 2-5
Symptom Summary
4 Look for duplicate problem.
Terminal Output
5 For an existing problem:
• Obtain the problem number from the user.
• Obtain any additional supporting data file information.
• Obtain any user free form comments to be added.
• Add information to the problem report.
Terminal Input
Return to CMS
PRBnnnnn Report
Terminal Prompt

Notes	Module	Label	Ref
3 The date and time stamps are supplied from CMS low storage. The FSWRITE routine writes the first two records, which contain general information about the problem. The third record, containing a start of keyword area indicator, is written.	DMMWRT	INTOUT	
• Keyword data is passed in variable blocked format. The data is extracted and moved to the output buffer one entry per 80-character record. FSWRITE adds this data and an end of keywords record.		KEYOUT	
• Supplementary data file names are added to the problem report file if supplied.		SUPPOUT	
• Textual descriptions of the problem are added to the file if supplied.		TEXTOUT	
• Data from INTSECT (the internal data area) low storage (time and date) and the initial status fields are moved to the 80-byte output area and FSWRITE adds the data to the symptom summary file.		CNTRLOUT	
• The keyword data is rounded up to a multiple of 80 bytes and the information is added to the symptom summary file.		CNTRLOUT	

Notes	Module	Label	Ref
4 Look for duplicate problem:			
• The keyword data for the new problem is compared to that of all existing problems and any exact matches are considered duplicates. The search is terminated when the newly created problem is encountered.	DMMSEA	START	
• The user is notified (at the terminal) as each duplicate problem is encountered. Up to 10 duplicate problems may be displayed for a search.		PUTOUT	
• If duplicate was found, DMMSUM is called to record the first encountered duplicate problem number in both the symptom summary control record for the problem, and the problem report.		ENDRTN	
5 A response of 'YES' to the prompt indicates that this is an update to an existing problem:	DMMPRO	OLDPROB	
• The user is prompted for the number of the problem, and its existence is verified.			
• The user is prompted for any additional data file names.		GETSDATA	
• The user is prompted for free form comments to be added to the report.		TEXTENTR	
• The new information, with a date and time stamp is added to the problem report.		OLDADD	

Diagram 2-3. PROB IPCS Command (Part 2 of 2)



1 Determine type of request.

2 STAT ALL with no other operands produces a file of current status for all problems.

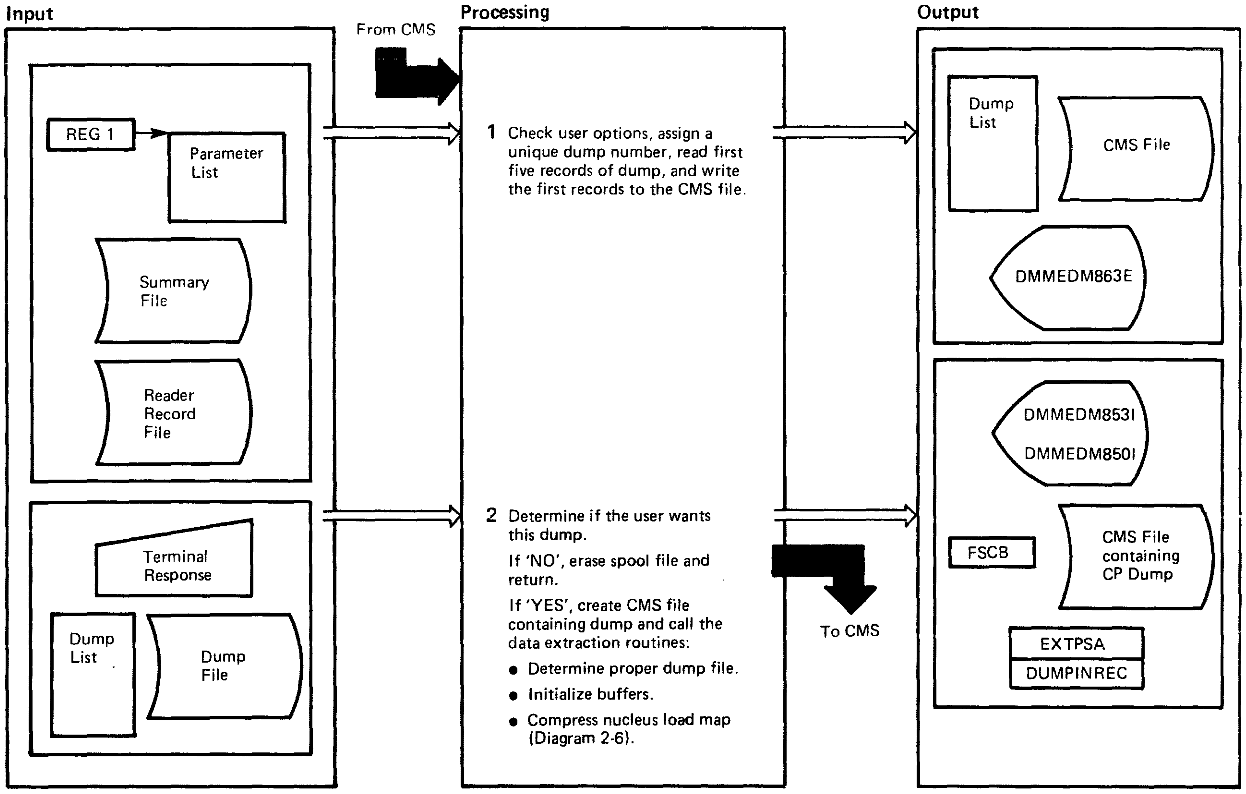
3 STAT ALL with other operands that request specific categories, produces a display of the requested status.

4 STAT with no operands causes a prompt of the user for problem number and a display of the specified status.

5 STAT nnnnn is a request for a display of the current status of the specified problem.

Notes	Module	Label	Ref
1 If an operand is not recognized, issue message: OPERAND NOT RECOGNIZED, STATALL ASSUMED	DMMSTA	START CK2CONT	
2 STAT ALL: Set switch (LALLSW) and erase any old copy of STATALL LOCAL file. Heading line is written followed by all the symptom summary control records and the file is closed.		STALLOC	
3 STAT ALL oper: If any additional operands are not recognized, issue message: OPERAND oper NOT RECOGNIZED • If operands are valid, the entire symptom summary file is searched and each control record is matched with the specified operands. • If a match is found, the control record is presented to the user on the terminal and the search continues.		STATSRCH	
4 STAT: The user is prompted for the number of the problem whose status he wishes. • If he enters 0000, STAT ALL is assumed (see Step 2). • If he enters a number other than 0000, that number is checked for validity and the symptom summary file is searched for the requested problem.		SPNUM	
5 STAT nnnnn: The problem number nnnnn is checked for validity. • If the number is in the correct format, the symptom summary file is searched for the requested problem. • The status is displayed when found. • If the problem is not found, issue message: PROBLEM NOT FOUND IN SYMPTOM SUMMARY FILE		STATRDY	

Diagram 2-4. STAT IPCS Command



1 Check user options, assign a unique dump number, read first five records of dump, and write the first records to the CMS file.

2 Determine if the user wants this dump.

If 'NO', erase spool file and return.

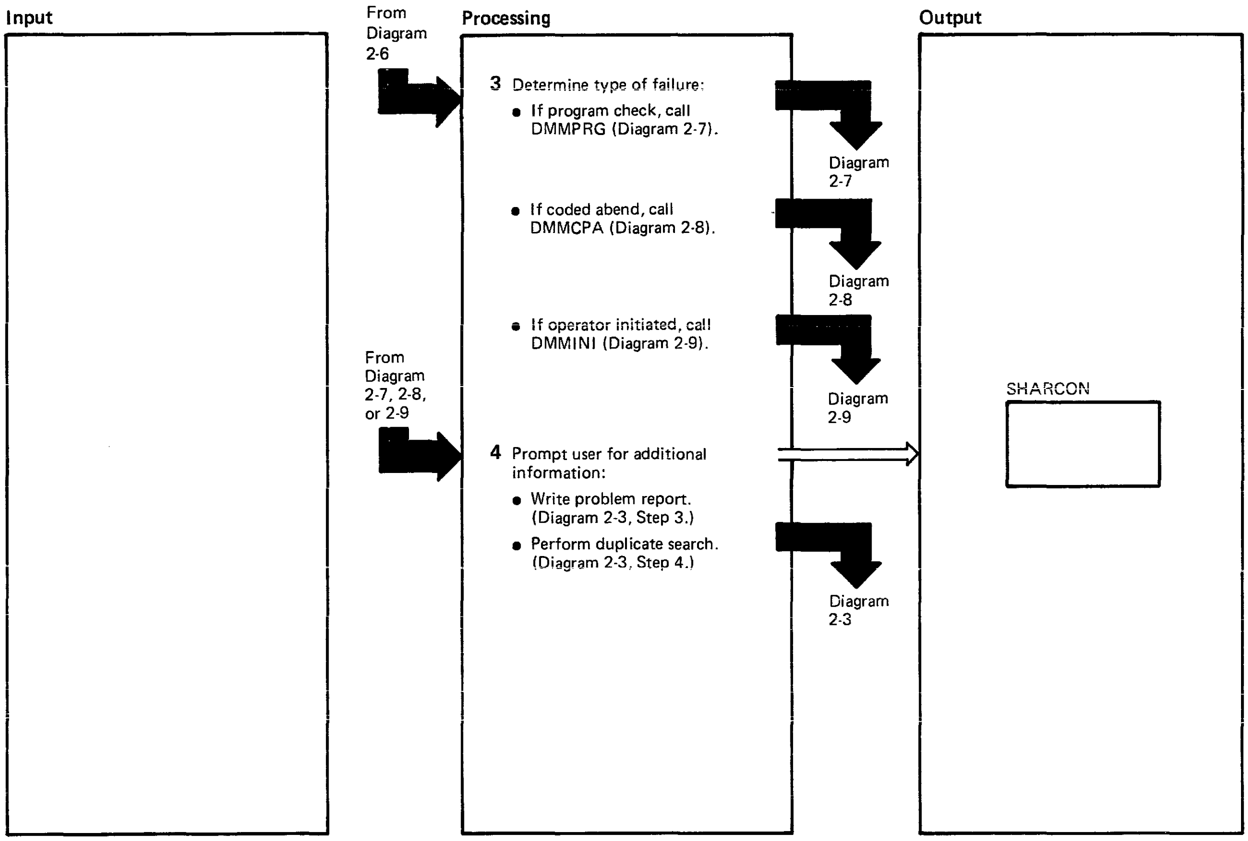
If 'YES', create CMS file containing dump and call the data extraction routines:

- Determine proper dump file.
- Initialize buffers.
- Compress nucleus load map (Diagram 2-6).

Notes	Module	Label	Ref
1 If there are no options specified, the defaults are MAP, FORMAT and HEX. If an invalid option is specified, issue message:	DMMEDM	CHKOPT	
INVALID PARAMETER parm		ERRFND	
PAGE REFERENCED NOT AVAILABLE	DMMINI	PRBDUMNO	
Read the next sequential number from the summary record file, and append it to the dump prefix 'PRB'. The dump name will always be PRBnnnnn, where 'nnnnn' is the unique dump number.	DMMEDM	RDUMP	
Read the records by branching and linking to the READCPR routine. The spool records contain the following: **Record** **Contents** 1 Symbol Table 2 Dump Information record 3-4 Storage Protection Keys at time of dump 5 First page of storage dumped (0) Save the following from the information record (2): • First 256 bytes of storage • General and floating-point and control registers • TOD clock and comparator • Address of the terminating prefix storage area • Abend code		NXTWD	
Create the item table from the bit map in record 2.		RDUMP	
Write the records to the CMS file that will contain the CP dump file (if requested) by branching and linking to the WTREC routine.			

Notes	Module	Label	Ref
2 To determine if the user wants this dump, issue the message: VM/370 SYSTEM ABEND xxxx DATE (date) TIME (time) DO YOU WANT THIS DUMP?			
If the user responds 'NO', erase the spool file and return to CMS.		REREAD	
If the user responds 'YES', write a record by branching and linking to the WRT Routine, read another record by branching and linking to the READCD routine, and so on until the read returns a non-zero condition code:		LOOP	
Code **Meaning** 1 End-of-file, 'DUMP PRBnnnnn CREATED' message is issued		DMPEND1	
2 Issue message 'NO DUMP FILES EXIST' and return to CMS		NODMP	
3 Issue message 'UNABLE TO READ DUMP FROM READER' and return to CMS		LOOP	
• Examine the dump list that resides in DMMEDM as a constant to determine the file name assigned to the dump; move this name to the read 'FSCB' to facilitate subsequent reads to the dump file.	DMMINI		
• Initialize buffers EXTPSA (terminating PSA in the dump) and EXTINREC (record 2 from the dump) by issuing FSREADs to the dump file.		EXTREAD	
• Call the nucleus load map module (DMMMAP) to compress the nucleus load map. (See Diagram 2-6.)		EXTREND	

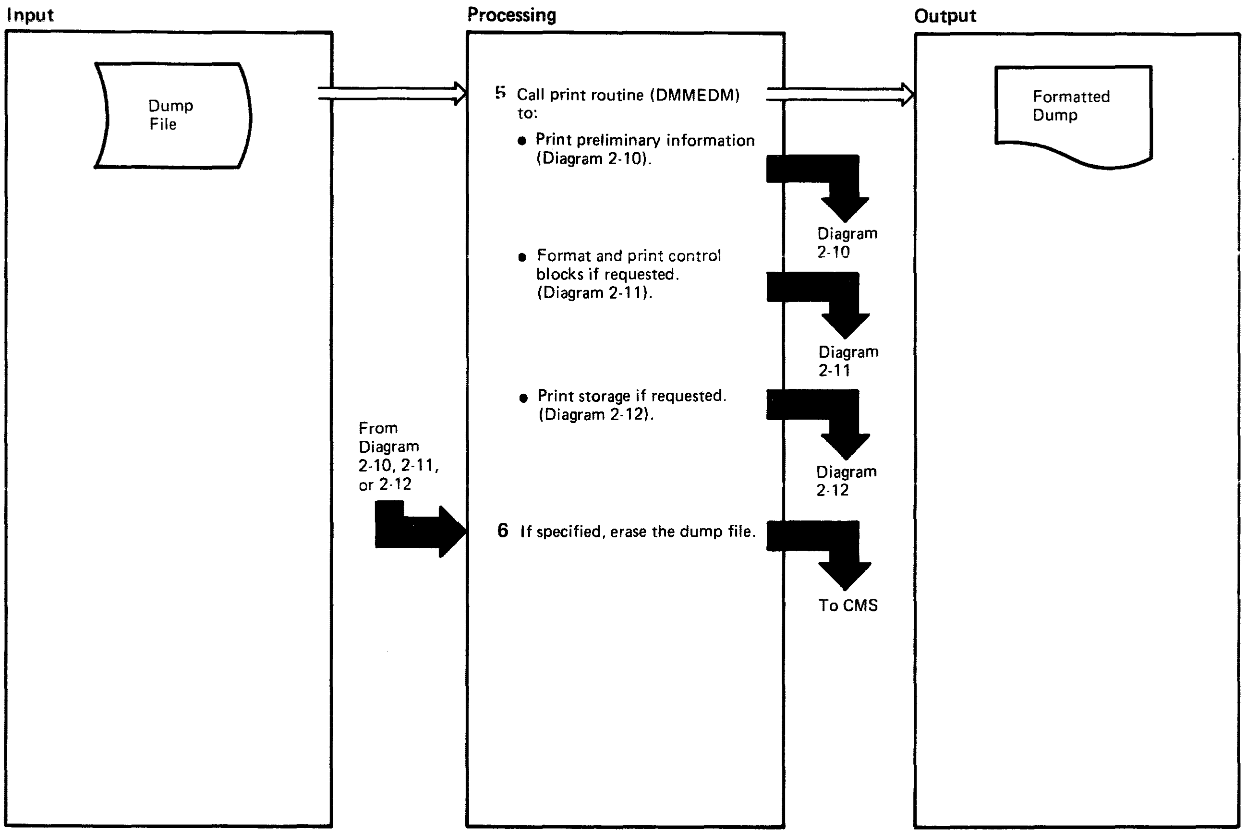
Diagram 2-5. VMFDUMP IPCS Command (Part 1 of 3)


Input
From Diagram 2-6
Processing
Output
3 Determine type of failure:
• If program check, call DMMPRG (Diagram 2-7).
Diagram 2-7
• If coded abend, call DMMCPA (Diagram 2-8).
Diagram 2-8
• If operator initiated, call DMMINI (Diagram 2-9).
Diagram 2-9
From Diagram 2-7, 2-8, or 2-9
SHARCON
4 Prompt user for additional information:
• Write problem report. (Diagram 2-3, Step 3.)
• Perform duplicate search. (Diagram 2-3, Step 4.)
Diagram 2-3

Notes	Module	Label	Ref
3 Examine the DMPABEND field of the dump information record (EXTINREC) to determine the failure type:	DMMINI	EXTPSWCK	
• If the failure type is a program check (PRGxx), call the program check routine (DMMPRG). See Diagram 2-7.			
• If the failure type is other than PSA02, call the coded abend routine (DMMCPA). See Diagram 2-8.		EXTSVCHK	
• If the failure type is PSA02, handle within DMMINI. See Diagram 2-9.		EXTPSCHK	
4 Prompt the user for the severity code,	DMMPRM	GETSEV	
examine the previously set switches in the SHARECON data area to determine the failure. If it was system detected (CP abend or program check), request the file name and file type of any supporting documentation, and a free form entry description of the problem.		PRMTYPSW PRMSUPP	
Note: The prompting sequence for operator initiated dumps depends on the user's response to the query: THE DUMP INFORMATION IS INCONCLUSIVE ENTER LOOP, PERFORMANCE OR OTHER		PRMLPPER	

Notes	Module	Label	Ref
• All information necessary to create the problem report has been gathered. Call module DMMWRT to order the data and create the problem report. See Diagram 2-3, Step 4.		NORMEXIT	
• Call module DMMSEA to search for duplicate problems. See Diagram 2-3, Step 4.			

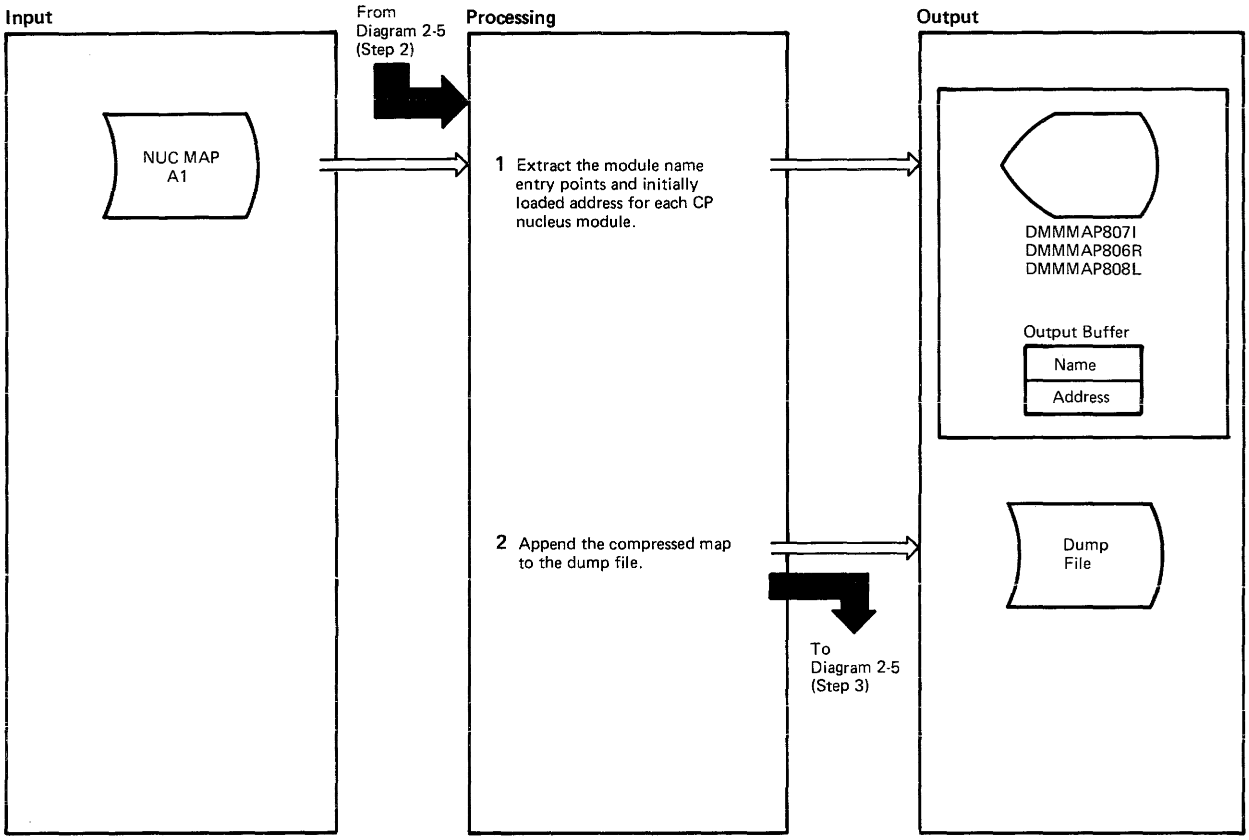
Diagram 2-5. VMFDUMP IPCS Command (Part 2 of 3)


Input
Processing
Output
Dump File
5 Call print routine (DMMEDM) to:
• Print preliminary information (Diagram 2-10).
Diagram 2-10
• Format and print control blocks if requested. (Diagram 2-11).
Diagram 2-11
• Print storage if requested. (Diagram 2-12).
Diagram 2-12
From Diagram 2-10, 2-11, or 2-12
6 If specified, erase the dump file.
To CMS
Formatted Dump

Notes	Module	Label	Ref
5 Pass control to the print routine (DMMEDM) to print the dump:	DMMPRM	EXIT	
• Read record1 (symbol table) and record2 (dump information) from the dump file and print the preliminary information. See Diagram 2-10.	DMMEDM	EDITDUMP	
• If the NOFORM option was omitted, format and print the control blocks. See Diagram 2-11.		RCHFORM	
• If the NOHEX option was omitted, print storage. See Diagram 2-12.		HEXDUMP	
6 If the ERASE option was omitted, keep the dump file. If specified, erase the dump file. In either case, return control to CMS.		RETN	

Notes	Module	Label	Ref

Diagram 2-5. VMFDUMP IPCS Command (Part 3 of 3)

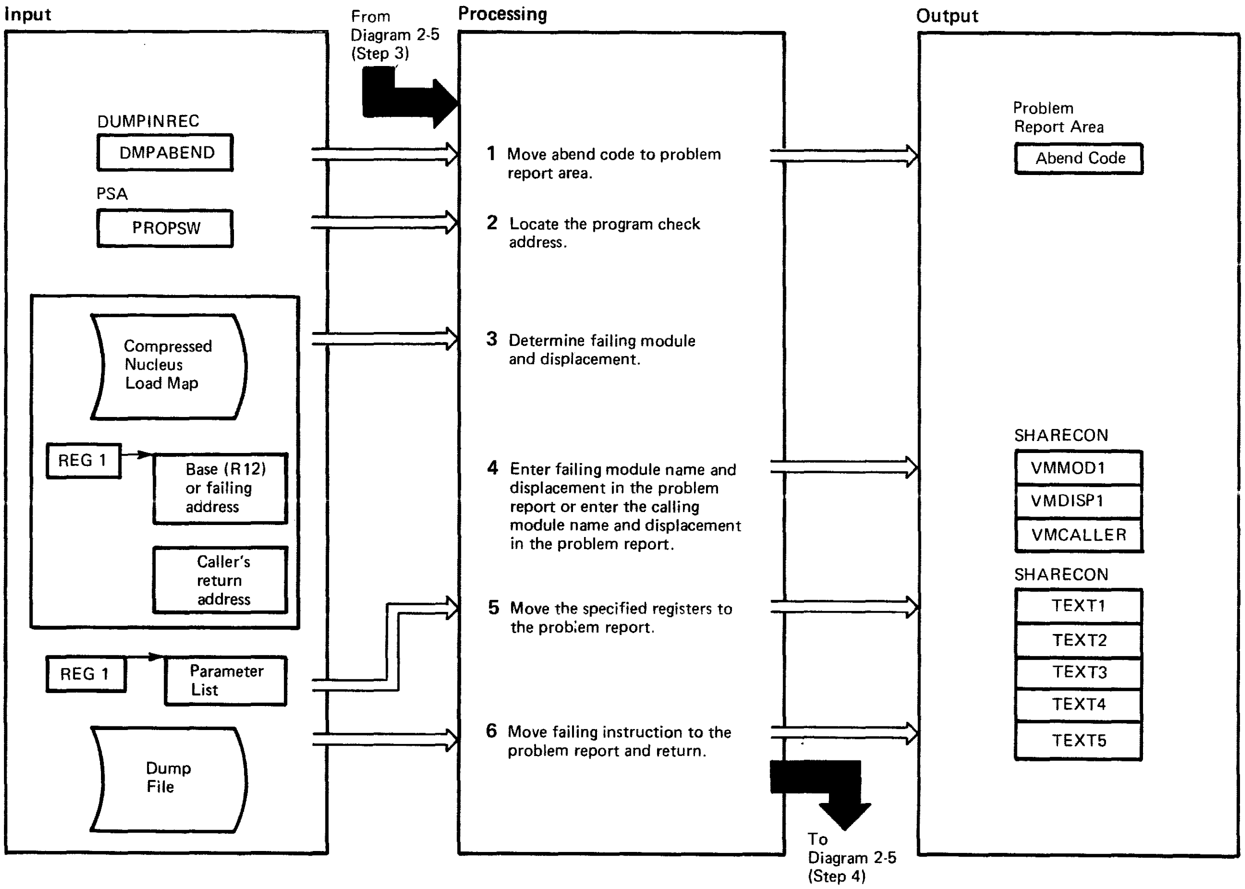


1 Extract the module name entry points and initially loaded address for each CP nucleus module.

2 Append the compressed map to the dump file.

Notes	Module	Label	Ref	Notes	Module	Label	Ref
1 Attempt a read to NUC MAP A1.	DMMAP	READ					
Note: The nucleus load map is assumed to reside on the IPCS user's A-disk. If NUC MAP A1 cannot be found, issue message:		READERR					
UNABLE TO LOCATE NUC MAP A1		STATERR					
Follow this message with message: ENTER fn ft fn OF THE NUCLEUS LOAD MAP		MAPNAME					
If the load map is successfully located, compare the address of the constant DMKCPEND in the symbol table (dump record1) to the address of DMKCPEND in the load map. If		XCK					
the addresses do not compare, issue message: NUCLEUS MAP INVALID 'file id'		MAPERROR					
If the map is valid, read each line of the map into a buffer. If it contains a module or entry point name, move this name and associated address (12 bytes) to an output buffer.		READ					
2 When end-of-file is reached sort the output buffer by ascending entry point address, write the output buffer and append it to the dump file.		WRTOUT					

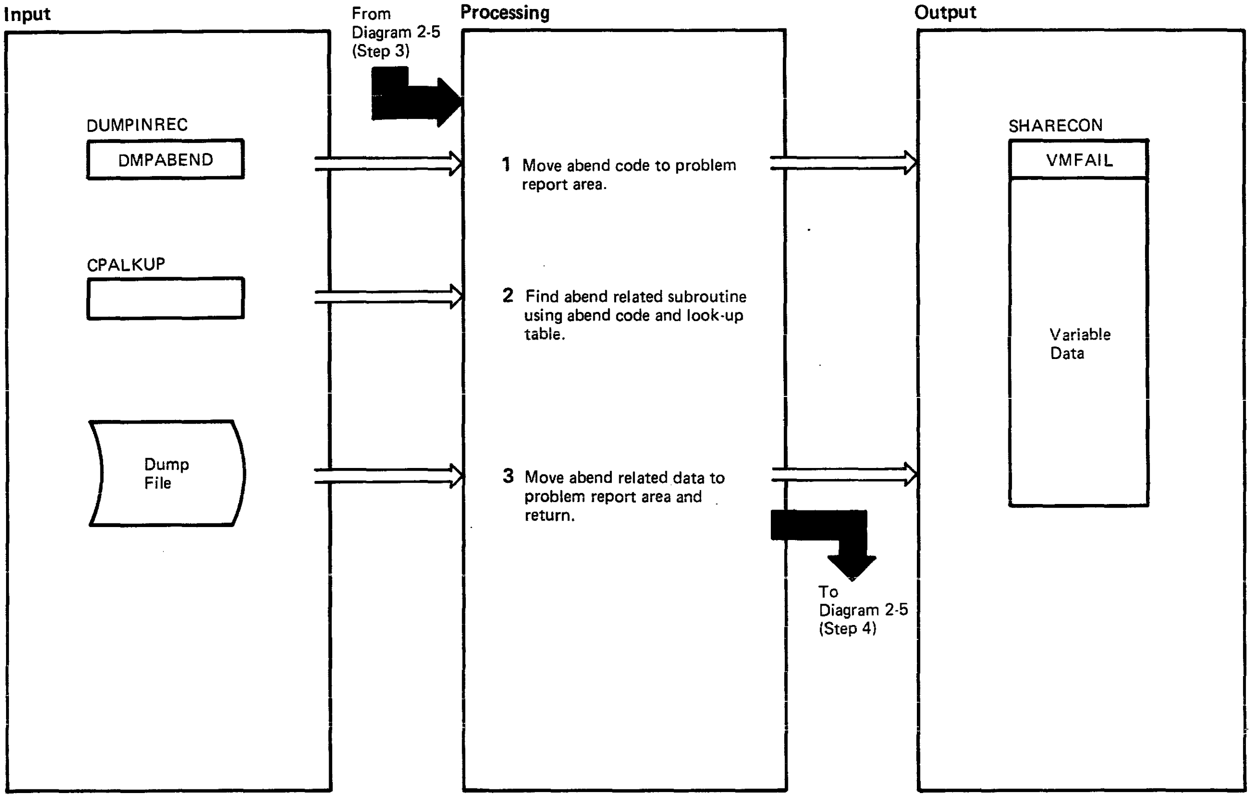
Diagram 2-6. Compress the Nucleus Load Map


Input
From Diagram 2-5 (Step 3)
Processing
Output
DUMPINREC
DMPABEND
PSA
PROPSW
Compressed Nucleus Load Map
REG 1
Base (R12) or failing address
Caller's return address
REG 1
Parameter List
Dump File
1 Move abend code to problem report area.
2 Locate the program check address.
3 Determine failing module and displacement.
4 Enter failing module name and displacement in the problem report or enter the calling module name and displacement in the problem report.
5 Move the specified registers to the problem report.
6 Move failing instruction to the problem report and return.
To Diagram 2-5 (Step 4)
Problem Report Area
Abend Code
SHARECON
VMMOD1
VMDISP1
VMCALLER
SHARECON
TEXT1
TEXT2
TEXT3
TEXT4
TEXT5

Notes	Module	Label	Ref
1 Move the abend code from DMPABEND into the problem report area.	DMMPRG		
2 Identify the program check address in the PSA Program Old PSW.			
3 Call DMMIDM to identify the failing module and displacement.			
If entered from DMMPRG, the failing address will be in the fixed nucleus portion of the dump or in a pageable module.	DMMIDM	MODREAL MODPRGCK	
If entered from DMMCPA, the caller's base (R12) will be in the fixed nucleus or in a pageable module.		MODABND MODPAGE	
Using the addresses provided, and the compressed nucleus map, calculate the displacement of the failing or calling module.			
4 Enter the name of the failing module in the problem report or enter the name of the calling module in the problem report.	DMMIDM	MODPRGCK MODGOOD	

Notes	Module	Label	Ref
5 Call DMMRMV to move the register set indicated by the pointer passed in register 1. It can be one of the following: ● general registers ● BALSAVE registers ● FREESAVE registers ● SAVEAREA registers ● LOKSAVE registers ● SWTSAVE registers	DMMRMV		
6 Move the failing instruction to the problem report and return to the user prompting routine.	DMMPRG	PTGINSTR PRGMORCD	

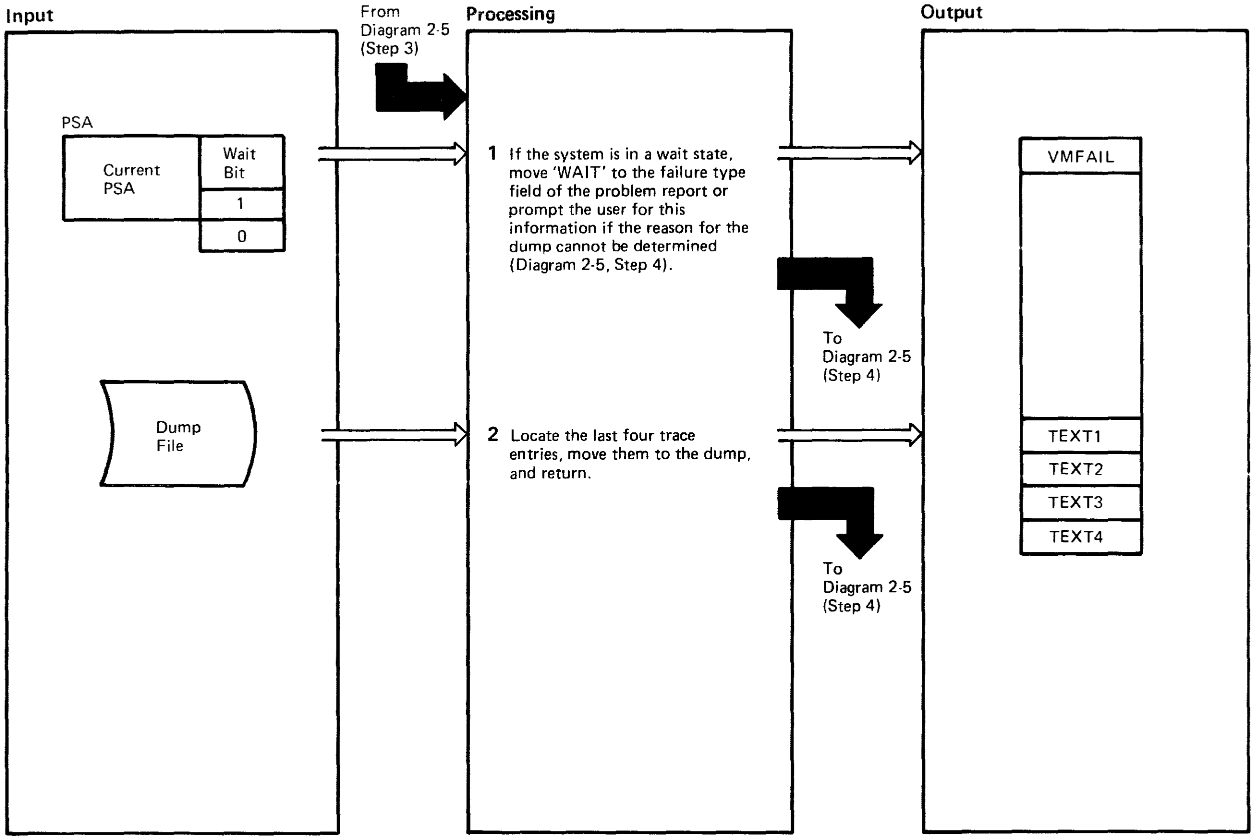
Diagram 2-7. Program Check Routine (DMMPRG)


Input
From
Diagram 2-5
(Step 3)
Processing
Output
DUMPINREC
DMPABEND
CPALKUP
Dump
File
1 Move abend code to problem report area.
2 Find abend related subroutine using abend code and look-up table.
3 Move abend related data to problem report area and return.
To
Diagram 2-5
(Step 4)
SHARECON
VMFAIL
Variable
Data

Notes	Module	Label	Ref
1 Move the abend code from the DMPABEND field of the dump information record (record 2) into the problem report area.	DMMCPA		
2 Compare the prefix of the abend code to the entries in the abend look up table; in this table an entry exists for every known abend and it includes a prefix code followed by a sub-routine address.		CPACNTIN	
3 When the appropriate subroutine is located, move the predetermined data to the problem report. *Note:* The amount and type of data extracted will vary according to the abend, but will generally fall into one of the following categories: ● TEXT 1 through TEXT 5 contain the IOBLOK. ● TEXT 1 through TEXT 5 contain an information type entry; for example, THE CLOCK IS IN ERROR OR NOT OPERATIONAL. ● TEXT 1 through TEXT 5 contain one of the six register save areas (reserved); VMDISP is the displacement within the caller; and VMCALLER is the module calling the failing module. The last category requires calling DMMIDM and DMMRMV. (See Diagram 2-7, Steps 4 and 5.)		CPAGETAD	
Call the user prompting routine.	DMMINI	CPACALL	

Notes	Module	Label	Ref

Diagram 2-8. Coded Abend Routine (DMMCPA)



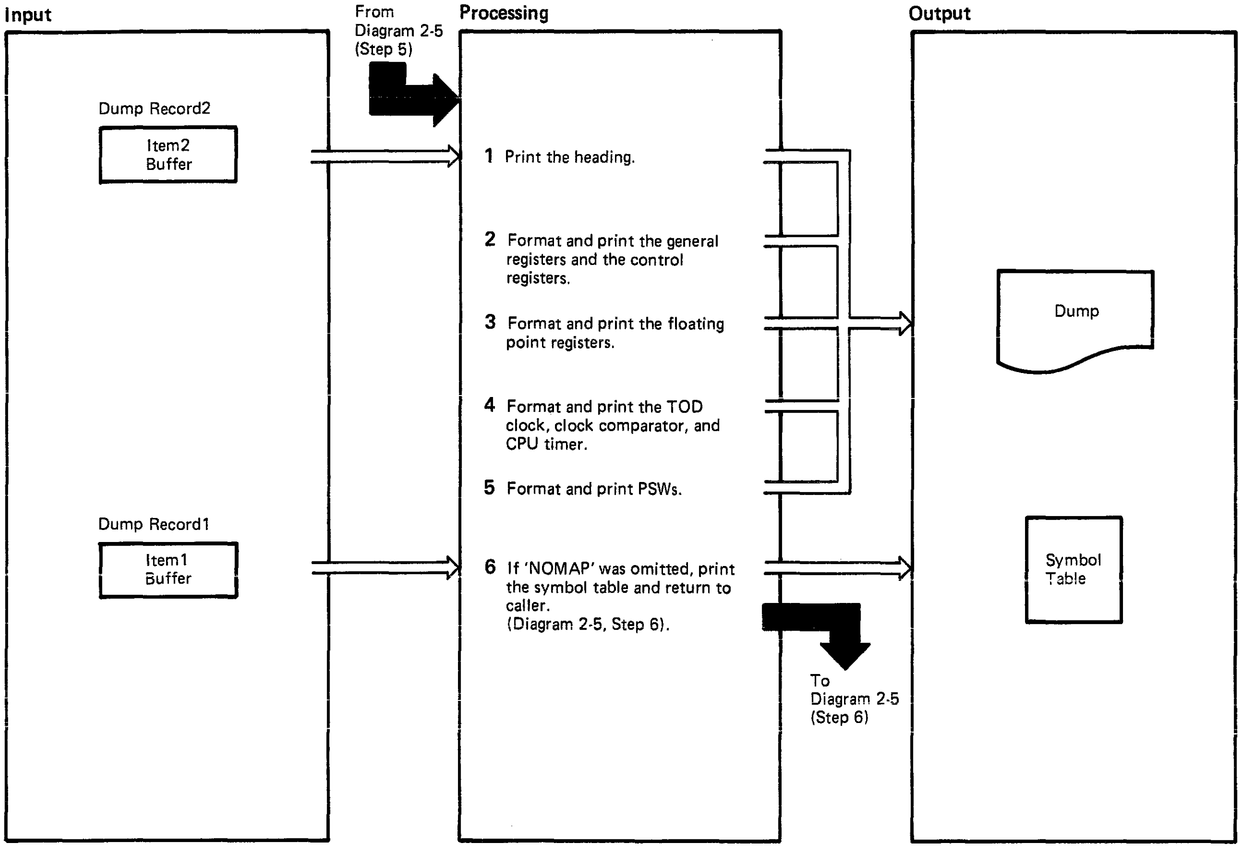
1 If the system is in a wait state, move 'WAIT' to the failure type field of the problem report or prompt the user for this information if the reason for the dump cannot be determined (Diagram 2-5, Step 4).

2 Locate the last four trace entries, move them to the dump, and return.

Notes	Module	Label	Ref
1 Examine the wait bit in the current PSW. If on, the system is assumed to be waiting when the operator depressed the SYSTEM RESTART key. Move WAIT to the problem report failure area and re-examine the current PSW to check for the presence of a wait code. If one exists, move it to the problem report also. If off, consider the dump information inconclusive and prompt the IPCS user for a failure code. (See Diagram 2-5, Step 4.)	DMMINI	EXTLPWT	
2 Locate the last four trace entries for all operator initiated dumps, move them to the problem report area, and return to the user prompting routine (Diagram 2-5, Step 4).		EXTTRTAB EXTLEAV	

Notes	Module	Label	Ref

Diagram 2-9. Operator Initiated Routine (DMMINI)



1 Print the heading.

2 Format and print the general registers and the control registers.

3 Format and print the floating point registers.

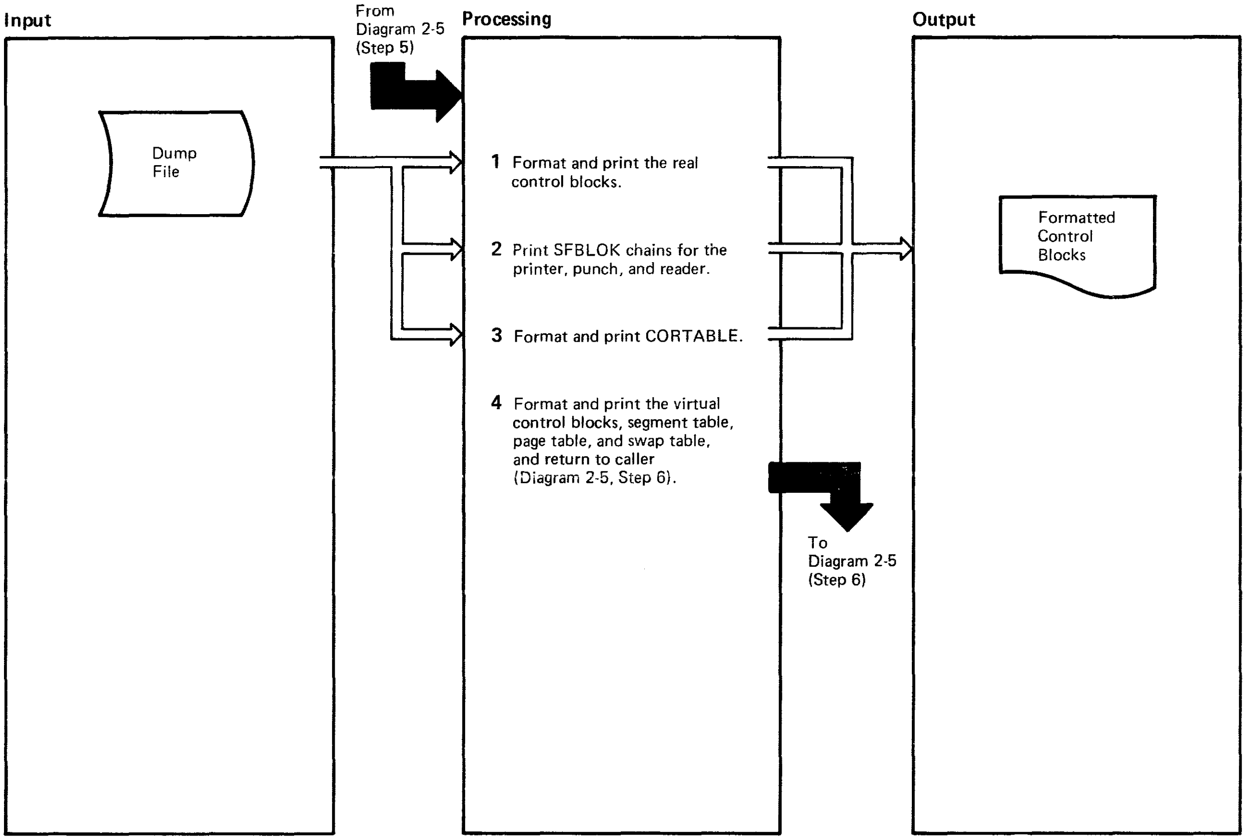
4 Format and print the TOD clock, clock comparator, and CPU timer.

5 Format and print PSWs.

6 If 'NOMAP' was omitted, print the symbol table and return to caller. (Diagram 2-5, Step 6).

Notes	Module	Label	Ref	Notes	Module	Label	Ref
1 Print the heading line which contains the time, date, abend code and cause.	DMMEDM	PREREC					
2 Unpack the general registers and the control registers by branching and linking to the transmit routine, move the data by branching and linking to the MVSBRTN routine and print it by branching and linking to the PRINTA routine.		PRELIM4					
3 Print the floating point registers as in Step 2, above.		PRELIM8					
4 Unpack and print the TOD clock, clock comparator and CPU timer as in Step 2, above.							
5 Translate low storage and format and print the PSWs.		PRELIM11					
6 If NOMAP was omitted, print the symbol table.							

Diagram 2-10. Print Preliminary Information (DMMEDM)



1 Format and print the real control blocks.

2 Print SFBLOK chains for the printer, punch, and reader.

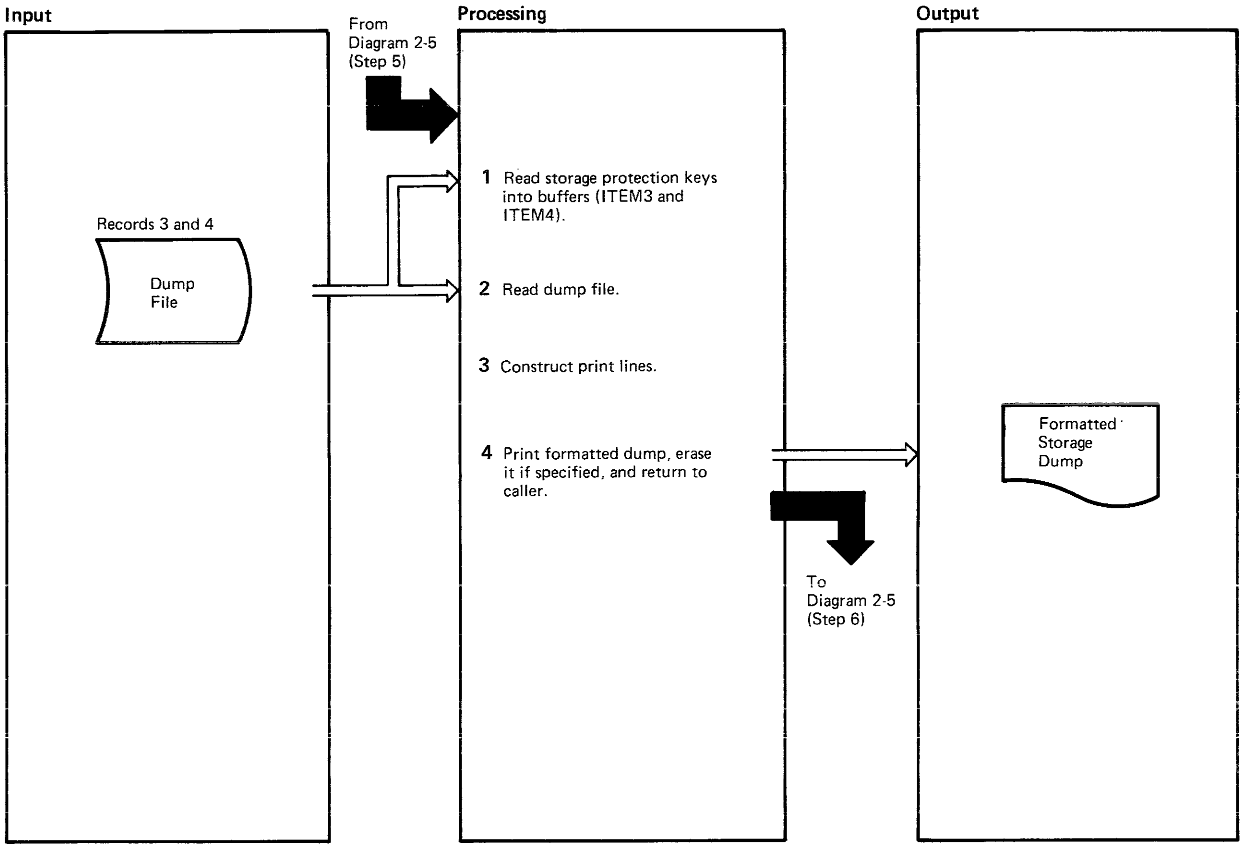
3 Format and print CORTABLE.

4 Format and print the virtual control blocks, segment table, page table, and swap table, and return to caller (Diagram 2-5, Step 6).

Notes	Module	Label	Ref
1 Format and print the following real control blocks:	DMMEDM	RCHFORM	
• RCHBLOKs and IOBLOKs chained to them		RCHPROC	
• RCUBLOKs and IOBLOKs chained to them		RCUINIT	
• RDEVBLOKs		RDEVINIT	
• Active IOBLOK			
• RSPLCTL and SFBLOK for unit record devices		IOPROC	
• CONTASK for termination			
• RECBLOKs for CP owned DASD devices			
Branch and link to the following routines for commonly used functions:			
GETPAGE To get a page of storage.			
TRANINIT To translate control blocks into printable form.			
BLKPRINT To print real control blocks.			
IOBPRINT To print IOBLOK.			
SFPRINT To print SFBLOK.			
IOERPRINT To print IOERBLOK.			
2 Print the printer, reader, and punch SFBLOK chains.		SPFORM	
Note: PRTSPL points to the punch spool and RDRSPL points to the reader spool.			
3 Unpack and print the CORTABLE.		CORTBL	

Notes	Module	Label	Ref
4 Format and print the following virtual control blocks:		VIRTUALM	
VMBLOKs		VMPRINT	
ECBLOKs (if any)			
VCHBLOKs		VCHINIT	
VCUBLOKs		VCUINIT	
VDEVBLOKs		VDVINIT	
Active IOBLOK (if any)			
VCONCTL (for console)			
VSPLCTL and SFBLOK (for unit record devices)		TSTSPOOL	
Segment, Page and Swap Tables		VMCK	
Note: These subroutines branch and link to subroutines to perform commonly used functions.			
Subroutine Function			
GETPAGE Get the page of storage containing the control block.			
TRANINIT Unpack control block for printing.			
BLKPRINT Print control block.			
SFPRINT Print SFBLOCK.			
IOERPROC Print IOERBLOCK.			
SEGPGTB Print segment, page, and swap tables.			

Diagram 2-11. Format and Print Control Blocks (DMMEDM)



1 Read storage protection keys into buffers (ITEM3 and ITEM4).

2 Read dump file.

3 Construct print lines.

4 Print formatted dump, erase it if specified, and return to caller.

Notes	Module	Label	Ref
1 DMMEDM reads record 3 and 4 from the dump file into the buffers called item3 and item4. These records contain the storage protection keys.	DMMEDM	HEXDUMP	
2 Read the remainder of the dump file, a page at a time, and place them in the print buffer.		READPAGE	
3 Construct the print line, placing the storage keys with the associated hexadecimal storage contents. (Printing of identical lines is suppressed.)		GETKEY	
4 Print the dump by branching and linking to the PNTPAGE routine. If ERASE is specified, erase it and issue message: DUMP PRBnnnnn PRINTED AND ERASED		GETKEY1	

Notes	Module	Label	Ref

Diagram 2-12. Print Storage (DMMEDM)

Program Organization

This section describes the program organization of Interactive Problem Control System (IPCS). The logic of modules DMMCPA, DMMDIR, DMMDSC, DMMEDM, DMMFED, DMMFEX, DMMGET, DMMGRC, DMMHEX, DMMIDM, DMMINI, DMMINT, DMMOIB, DMMLOC, DMMMAP, DMMMOD, DMMPRG, DMMPRM, DMMPRO, DMMREG, DMMRMV, DMMSCR, DMMSEA, DMMSTA, DMMSUM, DMMTRC, DMMTRN, DMMVMB, and DMMWRT.

DMMCPA -- EXTRACTS INFORMATION PERTINENT TO INDIVIDUAL ABEND CONDITIONS AND ENTERS IT IN A PROBLEM REPORT

Entry point

DMMCPA

Entry Conditicns

At entry, the shared constant area contains information previously gathered from the dump, and the PSA has been read into EXTPSA buffer.

Exit Conditions

The abend code and data related to that abend are in the problem report.

Routines Called

DMMIDM -- Which finds the calling module and displacement.
DMMTRN -- Which translates the data from hexadecimal to EBCDIC.
DMMGRC -- Which reads in the requested dump file records.
DMMPRM -- Which is the user prompting routine.
DMMRMV -- Which moves the registers to the problem report.

Called By

DMMINI

Error Messages

DMMCPA805I

DMMDIR -- FORMATS AND DISPLAYS HEXADECIMAL DATA ON THE TERMINAL SCREEN

Entry Point

DMMDIRLN -- Which displays the HELP pages for DMMDSC.
DMMDIR -- Which formats a screen from dump data.

Entry Conditions

R2: Points to the area to be displayed.
R7: The dump address to be displayed

Exit Conditions

R15: The return code
0 Good
4 Print error
8 Unrecoverable error

Routines Called

DMMINT -- Which translates from hexadecimal to EBCDIC.

Called By

DMMFEX, DMMLOC, DMMMOD, and DMMSCR

Error Messages

None

DMMDSC -- PROVIDES A METHCD OF EXAMINING THE CMS FORMAT CP DUMPS CREATED BY VMFDUMP

Entry Point

DUMPSCAN

Entry Conditions

From CMS when the DUMPSCAN command is issued.

Exit Conditions

R15: Return code:
0 User 'HX', 'QUIT', or 'END'
8 Error processing the dump

Routines Called

DMMFEX -- Which writes a full screen from the dump.
DMMFED -- Which displays areas of the dump.
DMMLOC -- Which locates data strings.
DMMSCR -- Which performs the scroll function.
DMMREG -- Which displays the registers.
DMMVMB -- Which displays the VMBLOK summary.
DMMMOD -- Which finds the modules and resolves the addresses.
DMMTRC -- Which displays the trace table entries.

Called By

CMS via the DUMPSCAN command.

Error Messages

DMMDSC700I
DMMDSC701R
DMMDSC719I
DMMDSC720I
DMMDSC721I
DMMDSC722I
DMMDSC723I

DMMEDM -- EDITS AND PRINTS A CP DUMP

Entry Point

DMMEDM

Entry Conditicns

R1:	Address of option list
R13:	SVC save area address
R14:	Return address
R15:	Entry point address

Exit Conditicns

If an error is encountered reading the CP dump file (register 15 is nonzero), refer to the CMS RDBUF code meanings.

Routines Called

RDBUF -- Via SVC to read in the dump file.
ERASE -- Via SVC to delete the CP dump file from the P-disk.
CLOSIO -- Via SVC to close out the printer.
PRINTR -- Via SVC to print a line on the printer.
TYPLIN -- Via SVC to write a message to the console.
DMMINI -- To create a problem report.

Called By

CMS via the VMFDUMP command.

Error Messages

DMMEDM100S
DMMEDM200S
DMMEDM400S
DMMEDM850I
DMMEDM852I
DMMEDM853I
DMMEDM860I
DMMEDM861E
DMMEDM863E
DMMEDM864I

DMMFED -- DISPLAYS 'NNN' BYTES FROM ADDRESS 'HEXLOC'

Entry Point

DMMFED -- Which formats the dump data.
DMMFEDLN -- Which writes a line to the terminal.

Entry Conditions

Register 2 points to the parameter list with input truncated to 8-byte words.

Exit Conditions

R15: Return code
- 0 Good
- 4 Error in DMMGET accompanied by message DMMFEX702I or DMMFEX703I
- 8 Unrecoverable error

Routines Called

DMMGET -- Which reads in an area of the dump.

Called By

DMMDSC, DMMFEX, DMMGET, DMMHEX, DMMOIE, DMMLOC, DMMMOD, DMMREG

Error Messages

DMMFED702I
DMMFED703I

DMMFEX -- DISPLAYS X'130' BYTES OF THE DUMP

Entry Point

DMMFEX

Entry Conditions

Register 2 points to the parameter list containing input truncated to 8-byte words.

Exit Conditions

R15: Return code
- 0 Good
- 4 Bad return from DMMGET or message DMMFEX704I is issued.

Routines Called

DMMDIR -- Which formats and displays the data.

Called By

DMMDSC, DMMLOC

Error Messages

DMMFEX704I

DMMGET -- FETCHES PORTIONS OF THE DUMP INTO STORAGE

Entry Point

DMMGET

Entry Conditions

Register 2 contains the required dump address.

Exit Conditions

Register 2 contains the requested area's in storage address.

R15: Return code
- 0 Good
- 4 Warning message DMMGET708I is issued

  R2 X'00' Page within dump not dumped

  R2 X'FF' Page outside range of dump
- 8 Read error and message DMKGET100S is issued

Routines Called

DMMINT -- Which translates the dump.

Called By

DMMFED, DMMFEX, DMMIOB, DMMLOC, DMMMOD, DMMREG, DMMSCR, DMMTRC, DMMVMB

Error Messages

DMMGET100S
DMMGET708I

DMMGRC -- READS DUMP RECORD CONTAINING DATA AT A GIVEN ADDRESS AND PASSES DATA BACK TO CALLER

Entry Point

DMMGRC

Entry Conditions

The shared constant area contains an address at GRCPARM which is the requested data address.

Exit Conditions

Under normal conditions, register 1 points to the data read from the dump. Upon error return, control is passed to DMMPRM to prompt the user for information before quitting.

Routines Called

DMMPRM -- Which handles errors if encountered (control not returned to DMMGRC).

Called By

DMMINI, DMMIDM, DMMCPA, DMMPRG

Error Messages

DMMGRC100S
DMMGRC809S

DMMHEX -- TRANSLATES EBCDIC TO HEXADECIMAL AND CHECKS FOR VALIDITY

Entry Point

DMMHEX

Entry Conditicns

R3: Ccntains the count in bytes.
R4: Points to the leftmost byte of EBCDIC.

Exit Conditicns

R5: Points to the leftmost byte of translated data.
R15: Return code
0 Good
4 Message DMMHEX714I is issued

Routines Called

DMMFED -- Which displays the dump data line by line.

Called By

DMMDSC, DMMIOB, DMMLOC, DMMMOD

Error Messages

DMMHEX714I

DMMIDM -- DETERMINES THE FAILING OR CALLING MODULE NAME AND DISPLACEMENT WITHIN THE MODULE

Entry Point

DMMIDM

Entry Conditicns

Register 1 points to a parameter list.
WORD 1 Failing address or base address of module.
WORD 2 Register 1 for the address of caller for non-program check condition.

Exit Conditions

The failing or calling module and displacement have been resolved and moved to the report. If possible, an entry point name is also determined.

Routines Called

DMMGRC -- Which reads the required record into the work buffer.
DMMTRN -- Which translates the displacement from binary to a printable format.

Called By

DMMCPA, DMMPRG

Error Messages

None

DMMINI -- INITIALIZES FOR DATA EXTRACTION FROM THE CMS FILE CONTAINING THE DUMP

Entry Point

DMMINI

Entry Conditions

Register 1 contains the VMFDUMP parameter list.

Exit Conditions

Normal exit is to DMMEDM to process the CP spool file.

If the problem number file (SUMMFILE) retrieval results in an error, control is returned to DMMEDM and VMFDUMP processing is halted.

Routines Called

DMMPRG -- Which extracts data for the CP program check.
DMMCPA -- Which extracts data for the CP coded abend.
DMMMAP -- Which compresses the load map.
DMMTRN -- Which translates the data from binary to zoned.
DMMGRC -- Which reads in the specified dump record.
DMMPRM -- Which prompts the user for additional problem information.

Called By

DMMEDM

Error Messages

DMMINI100S
DMMINI400S
DMMINI800S
DMMINI803S

DMMINT -- TRANSLATES THE BINARY DATA TO PRINTABLE FORMAT

Entry Point

DMMINT

Entry Conditicns

R3: Byte count
R4: Points to the input data string

Exit Conditicns

R5: Points to the translated data

Routines Called

None

Called By

DMMDSC, DMMDIR, DMMFED, DMMLOC, DMMMOD, DMMREG, DMMTRC, and DMMVMB

Error Messages

None

DMMIOB -- DISPLAYS THE I/O BLOCKS

Entry Point

DMMIOB

Entry Conditions

Register 2 pcints to the parameter list with the input truncated to 8-byte words.

Exit Conditicns

R15: Return code
- 0 Gccd
- 4 A bad return from DMMGET; message DMMIOB712I or DMMIOB713I was issued.
- 8 Unrecoverable error

Routines Called

CMMGET -- Which fetches data intc storage.
CMMHEX -- Which converts EBCDIC tc hexadecimal.

Called By

CMMCSC

Error Messages

CMMIOB712I
CMMIOB713I

DMMLOC -- LOCATES 'STRING' 'FRCM' 'TC' 'INCREMENT'

Entry Point

CMMLOC

Entry Conditions

Register 2 points to a parameter list containing the command.

Exit Conditions

R15: Return code
- 0 String is found and data is displayed.
- 4 Message DMMLCC715I is issued; no data is displayed.
- 8 Error in DMSFREE or DMSFRET; no data is displayed.

Routines Called

CMMGET -- Which fetches data into storage.
CMMINT -- Which translates hexadecimal into EBCDIC.
CMMHEX -- Which translates EBCDIC intc hexadecimal.
CMMFEX -- Which writes the found location to a screen.
CMMFEC -- Which writes the found location to a terminal.

Called By

CMMCSC

Error Messages

DMMLOC715I
DMMLOC716I
DMMLOC717I

DMMMAP -- APPENDS COMPRESSED AND SORTED LOAD MAP AT END OF DUMP FILE

Entry Point

DMMMAP

Exit Conditions

Register 1 points to the parameter list. Word 1 of the parameter list points to the 12K output buffer.

Exit Conditions

```
R15:    Return code
        0  Normal completion (the load
           map information is appended to
           the dump).
        8  Function not performed (error
           encountered).
```

Routines Called

None

Called By

DMMINI, DMMDSC

Error Messages

DMMMAP810S
DMMMAP200S
DMMMAP801I
DMMMAP802I
DMMMAP806R
DMMMAP807I
DMMMAP808I
DMMMAP810S

DMMMOD -- LOCATES MODULES AND ENTRY POINTS IN LOAD MAP AND IDENTIFIES MODULE CONTAINING GIVEN ADDRESS

Entry Point

DMMMOD

Entry Conditions

Register 2 points to the parameter list with input truncated to 8-byte words.

Exit Conditions

```
R15:    Return code
        0  Good
        4  Message DMMMOD705E,
           DMMMOD706I, DMMMOD707I or
           DMMMOD718I is issued.
        8  Unrecoverable error
```

Called By

DMMDSC

Error Messages

DMMMOD100I
DMMMOD705E
DMMMOD706I
DMMMOD707I
DMMMOD718I

DMMPRG -- HANDLES THE CP PROGRAM CHECK PROCESSING

Entry Point

DMMPRG

Entry Conditions

The shared constant area contains information about the failure.

Exit Conditions

Exits to DMMPRM (the prompting subroutine) with the failing code in the text area.

Routines Called

DMMRMV -- Which puts the registers in the output.
DMMGRC -- Which gets the dump record containing the code.
DMMTRN -- Which translates the failing code.
DMMPRM -- Which prompts the user for any information concerning the problem.

Called By

DMMINI

Error Messages

None

DMMPRM -- PROMPTS USER FOR SUPPLEMENTARY DATA FILES AND TEXTUAL NOTES ABOUT FAILURE

Entry Point

DMMPRM

Entry Conditicns

The common shared constant area contains informaticn gathered by previous routines.

Exit Conditions

Exit to DMMEDM with the problem report created, the symptom summary file appended, and the summary record updated.

Routines Called

DMMWRT -- Which writes the problem report to disk.
DMMSEA -- Which performs the duplicate problem search.

Called By

DMMINI, DMMCPA, DMMPRG

Error Messages

DMMPRM200S
DMMPRM804I

DMMPRO -- CREATES A PROBLEM REPORT THROUGH USER PROMPTING

Entry Point

DMMPRO

Entry Conditicns

Entry from CMS when PROB command is entered.

Exit Conditions

```
R15:    Return code
        0  Normal completion
        4  The user entered 'HX' (halt
           execution)
        8  Unrecoverable error
```

Routines Called

DMMWRT -- Which writes the problem report to disk.
DMMSEA -- Which looks for a duplicate of this problem.

Called By

By CMS when the PROB command is entered.

Error Messages

DMMPRO1COS
DMMPRO2COS

DMMREG -- DISPLAYS THE REGISTERS

Entry Point

DMMREG

Entry Conditions

Register 2 points to the parameter list with input truncated to 8-byte words.

Exit Conditions

```
R15:    Return code
        0  Good
        4  Warning
        8  Unrecoverable error
```

Routines Called

DMMINT -- Which translates hexadecimal to EBCDIC.
DMMFED -- Which displays the dump data line-by-line.
DMMGET -- Which fetches the dump pages into storage.

Called By

DMMDSC

Error Messages

DMMREG100S

DMMRMV -- PLACES REGISTERS IN THE TEXT AREA OF THE REPORT

Entry Point

DMMRMV

Entry Conditions

Register 1 points to the save area for one of the following savearea sets:

BALR
FREE
General registers
savearea

Exit Conditions

The registers are in the text area of the report.

Routines Called

DMMTRN -- Which translates the registers into a printable format.

Called By

DMMCPA, DMMPRG

Error Messages

None

DMMSCR -- SCROLLS THE DISPLAY UP OR DOWN FROM THE LAST ADDRESS

Entry Point

DMMSCR

Entry Conditions

Register 2 points to the parameter list containing the input truncated to 8-byte words.

Exit Conditions

```
R15:    Return code
        0  Good
        4  Message DMMSCR709I is issued;
           bad return from DMMGET.
        8  Unrecoverable error
```

Routines Called

DMMGET -- Which fetches data into storage.
DMMDIR -- Which formats and displays data.

Called By

DMMDSC

Error Messages

DMMSCR709I

DMMSEA -- LOCATES ANY PROBLEMS WHICH ARE DUPLICATES OF A NEWLY ENTERED PROBLEM

Entry Point

DMMSEA

Entry Conditions

Register 1 points to a parameter list as follows:

```
WD1   Pointer to the internal data area
WD2   Pointer to the keyword string (with
      length fields)
WD3   Pointer to the text area for this
      problem
```

Exit Conditions

```
R15:    Return code
        0  No duplicates found
        4  Duplicates found
        8  Unrecoverable error
           encountered
```

Routines Called

DMMSUM -- Which posts duplicate status of the problem to the summary control record.

Called By

DMMPRM, DMMPRO

Error Messages

DMMSEA100S

DMMSTA -- DISPLAYS THE STATUS OF A GIVEN PROBLEM OR GROUP OF PROBLEMS OR ALL PROBLEMS

Entry Point

DMMSTA

Entry Conditions

Register 1 points to the passed parameters:

Module name	length 8
PARM1 (ALL or PNUM)	length 8
PARM2 (SRCH ARG1)	length 8
PARM3 (SRCH ARG2)	length 8

Exit Conditions

R15: Return code
- 0 Normal return, function performed
- 4 Problem number not found in symptom summary
- 8 Unrecoverable error encountered

Routines Called

None

Called By

CMS via the STAT command.

Error Messages

DMMSTA100S
DMMSTA200S
DMMSTA601S

DMMSUM -- UPDATES OR FINDS SYMPTOM SUMMARY CONTROL RECORD FOR A GIVEN PROBLEM AND PASSES IT TO CALLER

Entry Point

SUMMARY

Entry Conditions

Register 1 contains the parameter pointer as follows:

8 characters not used

8 characters PRBxxxxx, where xxxxx is the problem number

8 characters The function to be performed (UPcccccc) or FI where cccccc is EG, STAT, PTF, or FUNCT, and UP and FI stand for update and find

n characters New data to be put in the appropriate field

Exit Conditions

R15: Return code
- 0 Normal successful completion
- 4 Requested problem not found
- 8 Unrecoverable error encountered

If a FIND was requested, register 1 points to the problem control record.

Routines Called

None

Called By

DMMSEA and PRB EXEC

Error Messages

DMMSUM100S
DMMSUM200S
DMMSUM501S
DMMSUM502S

DMMTRC -- DISPLAYS 'NNN' TRACE ENTRIES

Entry Point

DMMTRC

Entry Conditions

Register 2 points to the parameter list.

Exit Conditions

Register 2 contains the last displayed scroll address.

R15: Return code
- 0 Good
- 4 Bad return from DMMGET, message DMMTRC710I is issued.

Routines Called

DMMGET -- Which fetches the dump pages into storage.
DMMFED -- Which displays the specified areas.

Called By

DMMDSC

Error Messages

DMMTRC710I

DMMTRN -- TRANSLATES BINARY DATA INTO A PRINTABLE FORMAT

Entry Point

DMMTRN

Entry Conditions

The common constant area TRNPARM and TRNPARM1 have the data length and data address respectively.

Exit Conditions

Register 1 points to the translated data.

Routines Called

DMMINI, DMMCPA, DMMPRG, DMMRMV, and DMMIDM

Error Messages

None

DMMVMB -- DISPLAYS ALL VMBLOK ADDRESSES, USERIDS, AND STATUS

Entry Point

DMMVMB

Entry Conditions

None

Exit Conditions

R15: Return code
- 0 Good
- 4 Bad return from DMMGET, message DMMVMB711I issued.

Routines Called

DMMGET -- Which fetches the dump into storage.
DMMINT -- Which translates hexadecimal into EBCDIC.

Called By

DMMDSC

Error Messages

DMMVMB711I

DMMWRT -- CREATES A PROBLEM REPORT ON DISK AND ADDS THIS PROBLEM TO THE SYMPTOM SUMMARY FILE

Entry Point

DMMWRT

Entry Conditions

Register 1 points to the parameter list as follows:

WD1	Points to the internal data (DSECT INTSECT)
WD2	Points to the keyword data (variable blocked format)
WD3	Points to the text description (halfword length prefix)
WD4	Points to the supplementary data (halfword length prefix)

Exit Conditions

R15: Return code
- 0 Normal, successful completion
- 8 Error occurred

Routines Called

None

Called By

DMMPRO (PROB command), and DMMPRM (during VMFDUMP processing)

Error Messages

DMMWRT200S

Directory

Figure 2-2 is an alphabetical list of some of the labels in the IPCS modules. The function performed at the point in the program indicated by each label is described and the associated method of operation diagram is referenced.

Label	Module	Diagram	Description
CPACALL	DMMCPA	2-8	Calls the user prompt routine.
CPACNTIN	DMMCPA	2-8	Scans the abend look up table.
CPALKUP	DMMCPA	2-8	Codes the abend look up table.
CHECKTWO	DMMDSC	2-1	Turns print on and off.
CLOSEPRT	DMMDSC	2-1	Issues the DIAGNOSE X'08' subcommand to close print.
ENTER	DMMDSC	2-1	Prompts for the dump name and file type.
FOUNDAMP	DMMDSC	2-1	Calls an entry from the &NAME table.
GETDUMP	DMMDSC	2-1	Prompts the user who has asked for HELP.
INCHECK	DMMDSC	2-1	Determines if the entry is HELP, QUIT, HX or END.
MAPCHECK	DMMDSC	2-1	Ensures that the dump has no map.
NOTINTAB	DMMDSC	2-1	Adds the entry to the &NAME table.
PLISTSCN	DMMDSC	2-1	Examines the parameter to determine what subroutine is needed.
READIN	DMMDSC	2-1	Issues an RDTERM to accept subcommands.
RESUBCOM	DMMDSC	2-1	Reissues the previous subcommand.
SHOWPRSW	DMMDSC	2-1	Displays the print status.
SHOWTAB	DMMDSC	2-1	Displays a list of table entries.
SUBCOM	DMMDSC	2-1	Issues the print subcommand.
TOKEN	DMMDSC	2-1	Groups the input in an 8-byte parameter list.
CHKOPT	DMMEDM	2-5	Checks the user options for accuracy.
CORTBL	DMMEDM	2-11	Edits and prints the storage table.
DMPEND1	DMMEDM	2-5	Issues the end-of-file message.
EDITDUMP	DMMEDM	2-5	Prints preliminary information.
ERASE	DMMEDM	2-5	Erases the CMS file containing the dump.

Figure 2-2. The Interactive Problem Control System (IPCS) Label Directory (Part 1 of 5)

Label	Module	Diagram	Description
ERROP	DMMEDM	2-5	Issues message DMMEDM863I.
ERROR3	DMMEDM	2-5	Issues message DMMEDM861I.
GETKEY	DMMEDM	2-12	Constructs the print line showing the storage keys.
GETKEY1	DMMEDM	2-12	Prints a storage line.
HEXDUMP	DMMEDM	2-5	Prints all of storage.
IOBPROG	DMMEDM	2-11	Formats and prints the IOBLOK.
LOADMAP	DMMEDM	2-10	Ensures that the symbol table is requested.
LOOP	DMMEDM	2-5	Creates a CMS file.
NODMP	DMMEDM	2-5	Issues message DMMEDM853I.
NXTWD	DMMEDM	2-5	Creates the item table from the bit map.
PRELIMII	DMMEDM	2-10	Changes the PSWs.
PRELIM4	DMMEDM	2-10	Prints the general and control registers.
PRELIM8	DMMEDM	2-10	Prints the floating proint registers.
PREREC	DMMEDM	2-10	Prints the heading.
RCHFORM	DMMEDM	2-5	Prints the real control blocks.
RCHPROC	DMMEDM	2-11	Prints RCHBLOKs.
RCUINIT	DMMEDM	2-11	Prints RCUBLOKs.
RDEVINIT	DMMEDM	2-11	Prints the RDEVBLOKs.
RDUMP	DMMEDM	2-5	Writes the CMS file containing the CP dump.
READPAGE	DMMEDM	2-12	Reads the dump file storage pages.
REREAD	DMMEDM	2-5	Reads the operator response.
RETN	DMMEDM	2-5	Saves the file if ERASE is not specified.
SETEDM	DMMEDM	2-5	Opens the dump file.
SFFORM	DMMEDM	2-11	Prints the SFBLOK chains for unit record I/O.

Figure 2-2. The Interactive Problem Control System (IPCS) Label Label Directory (Part 2 of 5)

Label	Module	Diagram	Description
TSTSPOOL	DMMEDM	2-11	Prints VSPLCTL and SFBLOKs.
VCHINIT	DMMEDM	2-11	Formats and prints VCHBLOKs.
VCUINIT	DMMEDM	2-11	Formats and prints VCUBLOKs.
VINIT	DMMEDM	2-11	Formats and prints VDEVBLOKs.
VIRTUALM	DMMEDM	2-11	Prints virtual control blocks.
VMCK	DMMEDM	2-11	Prints segment, page, and swap tables.
VMPRINT	DMMEDM	2-11	Formats and prints VMBLOKs.
CTRAN	DMMFED	2-1	Converts the count to hexadecimal.
DIRECT	DMMFED	2-1	Formats the dump data.
SCRNFULL	DMMFED	2-1	Writes the dump data.
TRANADD	DMMFED	2-1	Converts the address to hexadecimal.
DIRECT	DMMFEX	2-1	Calls DMMDIR to display the dump data.
INDIR	DMMFEX	2-1	Checks for indirect requests.
NOROUND	DMMFEX	2-1	Calls DMMGET to get the dump data.
TRANADD	DMMFEX	2-1	Converts the address to hexadecimal.
MODREAL	DMMIDM	2-7	Determines the address location.
EXTLEAV	DMMINI	2-9	Processes the operator initiated dump.
EXTLPWT	DMMINI	2-9	Examines wait bit in PSW.
EXTPSCHK	DMMINI	2-9	Processes operator initiated dump.
EXTPSWCK	DMMINI	2-5	Finds the failure type.
EXTREAD	DMMINI	2-5	Initializes the buffers for DUMPINREC.
EXTREND	DMMINI	2-5	Calls the map compression routine.
EXTSVCHK	DMMINI	2-5	Calls the coded abend routine.
EXTTRTAB	DMMINI	2-9	Locates the last four trace entries.
COMPRCUB	DMMIOB	2-1	Gets addresses for the real control blocks.
VIO	DMMIOB	2-1	Gets the addresses for the virtual control block.
DMMLOC	DMMLOC	2-1	Fetchs the 'from' page.
EXECUTOR	DMMLOC	2-1	Compares the string to the dump.
GOGOFEX	DMMLOC	2-1	Displays the equal compare.
MAPERROR	DMMMAP	2-6	Issues message DMMMAP808I.

Figure 2-2. The Interactive Problem Control System (IPCS) Label Directory (Part 3 of 5)

Label	Module	Diagram	Description
MAPNAME	DMMMAP	2-6	Issues message DMMMAP806R.
READ	DMMMAP	2-6	Reads in 'NUC MAP A'.
READERR	DMMMAP	2-6	Issues message DMMMAP807I.
WRTOUT	DMMMAP	2-6	Adds the compressed nucleus map to the dump.
XCK	DMMMAP	2-6	Checks the nucleus load map for validity.
DMMMOD	DMMMOD	2-1	Reads the load map.
MAPRED	DMMMOD	2-1	Scans the load map.
PAGEMOD	DMMMOD	2-1	Checks for a pageable module.
QREQUEST	DMMMOD	2-1	Scans the load map.
READ	DMMMOD	2-1	Reads the load map.
PRGMORCD	DMMPRG	2-7	Calls the user prompt routine.
EXIT	DMMPRM	2-5	Returns to the VMFDUMP print routine.
NORMEXIT	DMMPRM	2-5	Calls the write and search routines.
EXIST	DMMPRO	2-3	Prompts to determine new or old problem.
GETFAIL	DMMPRO	2-3	Prompts for specific type of problem.
GETSDATA	DMMPRO	2-3	Gets supporting data file names.
MAINLINE	DMMPRO	2-3	Gathers general problem data.
OLDADD	DMMPRO	2-3	Appends information to the problem report.
OLDPROB	DMMPRO	2-3	Gets the number of the old problem.
SRCHRTN	DMMPRO	2-3	Calls DMMSEA for a duplicate search.
TEXTENTR	DMMPRO	2-3	Prompts for free form text information.
DMMREG	DMMREG	2-1	Checks for AP or UP dump.
DIRECTIT	DMMSCR	2-1	Formats and displays.
DMMSCR	DMMSCR	2-1	Calculates new display address.
ENDRTN	DMMSEA	2-3	Updates symptom summary with duplicate entry.
PUTOUT	DMMSEA	2-3	Notifies user of duplicate problem.
START	DMMSEA	2-3	Searches for a duplicate problem.
SPNUM	DMMSTA	2-4	Prompts for the problem number.

Figure 2-2. The Interactive Problem Control System (IPCS) Label Directory (Part 4 of 5)

Label	Module	Diagram	Description
STALLOC	DMMSTA	2-4	Creates the STATALL LOCAL file.
START	DMMSTA	2-4	Checks the first operand for validity.
STATRDY	DMMSTA	2-4	Displays the status of a given problem.
STATSRCH	DMMSTA	2-4	Searches for the type of status record.
REPORT1	DMMSUM	2-2	Appends status change to problem report.
START	DMMSUM	2-2	Determines the type of request.
DMMTRC	DMMTRC	2-1	Finds the trace table.
BALGET	DMMVMB	2-1	Chains through the VMBLOKs.
DMMVMB	DMMVMB	2-1	Gets the system VMBLOK pointer.
MOVEL	DMMVMB	2-1	Prints the VMBLOK list.
CNTRLOUT	DMMWRT	2-3	Adds PROB control record to the symptom summary.
INTOUT	DMMWRT	2-3	Writes the first three report records.
KEYOUT	DMMWRT	2-3	Adds keyword data to the report.
SUPPOUT	DMMWRT	2-3	Adds supplementary file names to report.
TEXTOUT	DMMWRT	2-3	Adds free form text to report.
-APAR	PRB EXEC	2-2	Posts the PARM number.
-CLOSE	PRB EXEC	2-2	Closes the problem.
-DUPOF	PRB EXEC	2-2	Posts the problem as a duplicate.
-IBM	PRB EXEC	2-2	Posts the report to IBM.
-MORE	PRB EXEC	2-2	Indicates that more information is needed.
-PTFIS	PRB EXEC	2-2	Posts the PTF number.
-PTFON	PRB EXEC	2-2	Applies the posted PTF.
-RETRY	PRB EXEC	2-2	Checks the operand for validity.
-SEV	PRB EXEC	2-2	Changes the severity.
-USER	PRB EXEC	2-2	Posts the problem as the user's responsibility.

Figure 2-2. The Interactive Problem Control System (IPCS) Label Directory (Part 5 of 5)

Data Areas

This section describes the data areas used by the Interactive Problem Control System (IPCS). The data areas are:

SHARECON -- VMFDUMP SHARED CONSTANT AREA

```
    ┌───────────────────────────────────────────────────────────────┐
 00 │  TYPWSW      │      P            R            B              │
    │───────────────────────────────────────────────────────────────│
 04 │  NUM                                                          │
    │              ┌────────────────────────────────────────────────│
 08 │              │                   *                            │
    │───────────────────────────────────────────────────────────────│
 0C │                        DMPFSCB                                │
    │───────────────────────────────────────────────────────────────│
 10 │                        GRCPARM                                │
    │───────────────────────────────────────────────────────────────│
 1  │                        TRNPARM                                │
    │───────────────────────────────────────────────────────────────│
 18 │                        TRNPARM1                               │
    │───────────────────────────────────────────────────────────────│
 1C │                        MODDISP                                │
    │───────────────────────────────────────────────────────────────│
 20 │                        REALEND                                │
    │───────────────────────────────────────────────────────────────│
 24 │                        EDMRET                                 │
    │───────────────────────────────────────────────────────────────│
 28 │                        EDMSAVE                                │
    <                                                               >
    <                                                               >
    │───────────────────────────────────────────────────────────────│
 68 │                        KEY                                    │
    │───────────────────────────────────────────────────────────────│
 6C │                        COMPLN                                 │
    │───────────────────────────────────────────────────────────────│
 70 │                        VMCOMPID                               │
    │                                                               │
 74 │                                                               │
    │              ┌────────────────────────────────────────────────│
 78 │              │             VMCOMP1                            │
    │──────────────┘                                                │
 7C │                                                               │
    │───────────────────────────────────────────────────────────────│
 80 │                            │           VMENVL                 │
    │───────────────────────────────────────────────────────────────│
 84 │                            │           VMENV                  │
    │────────────────────────────┘                                  │
 88 │                                                               │
    │                                                               │
 8C │                                                               │
    │───────────────────────────────────────────────────────────────│
 90 │                        PLCLN                                  │
    │───────────────────────────────────────────────────────────────│
 94 │                        VMPLC                                  │
    │                            ┌──────────────────────────────────│
 98 │                            │          VMPLC1                  │
    │───────────────────────────────────────────────────────────────│
 9C │             │                  SCPLN                          │
    └───────────────────────────────────────────────────────────────┘
```

Figure 2-3. VMFDUMP Shared Constant Area (Part 1 of 4)

```
     +-----------------------------------------------------------+
  A0 |             |              VMSCPLV                        |
     |-------------+                                             |
  A4 |                                                           |
     |             +---------------------------------------------|
  A8 |             |                VMSCPLV1                     |
     |-----------------------------------------------------------|
  AC |                        FAILLN                             |
     |-----------------------------------------------------------|
  B0 |                        VMFAIL                             |
     |                                                           |
  B4 |                                                           |
     |                            +------------------------------|
  B8 |                            |          VMFAIL1             |
     |----------------------------+                              |
  BC |                                                           |
     |                                                           |
  C0 |                                                           |
     |             +---------------------------------------------|
  C4 |             |                MODLN                        |
     |-----------------------------------------------------------|
  C8 |             |                VMMOD                        |
     |-------------+                                             |
  CC |                                                           |
     |                            +------------------------------|
  D0 |                            |          VMMOD1              |
     |----------------------------+                              |
  D4 |                                                           |
     |-----------------------------------------------------------|
  D8 |                        DISPLN                             |
     |-----------------------------------------------------------|
  DC |                        VMDISP                             |
     |                                          +----------------|
  E0 |                                          |   VMDISP1      |
     |-----------------------------------------------------------|
  E4 |                                          |   ENTRYLN      |
     |-----------------------------------------------------------|
  E8 |                                          |   VMENTRY      |
     |------------------------------------------+                |
  EC |                                                           |
     |                                          +----------------|
  F0 |                                          |   VMENTRY1     |
     |------------------------------------------+                |
  F4 |                                                           |
     |                                          +----------------|
  F8 |                                          | 80    *        |
     |-----------------------------------------------------------|
     <                                                           >
     |-----------------------------------------------------------|
 14C |                        SUPPORT                            |
     |-----------------------------------------------------------|
 150 |           SUPPLN             |            SUPP1           |
     |------------------------------+                            |
     <                                                           >
     <                              +----------------------------->
 1A0 |                              |            SUPP2           |
     |------------------------------+                            |
     <                                                           >
     <                              +----------------------------->
 2E0 |                              |            TEXT            |
     +-----------------------------------------------------------+
```

Figure 2-3. VMFDUMP Shared Constant Area (Part 2 of 4)

```
    ┌─────────────────────────────────────────────────────────────┐
2E4 |                           TEXTA                             |
    <                                                             >
    <                                                             >
    |─────────────────────────────────────────────────────────────|
334 |                           TEXT1                             |
    <                                                             >
    <                                                             >
    |─────────────────────────────────────────────────────────────|
384 |                           TEXT2                             |
    <                                                             >
    <                                                             >
    |─────────────────────────────────────────────────────────────|
3D4 |                           TEXT3                             |
    <                                                             >
    <                                                             >
    |─────────────────────────────────────────────────────────────|
424 |                           TEXT4                             |
    <                                                             >
    <                                                             >
    |─────────────────────────────────────────────────────────────|
474 |                           TEXT5                             |
    <                                                             >
    <                                                             >
    └─────────────────────────────────────────────────────────────┘
```

Displacement Hex	 Dec	Field Name			Description
00	00	TYPESW	DC	X'00'	Dump type switch
		WAITSW	EQU	X'01'	Wait
		LOOPSW	EQU	X'02'	Loop or performance
		PRGCKSW	EQU	X'04'	Program check
		CPABSW	EQU	X'08'	CP coded abend
		PROCERR	EQU	X'80'	Data extraction process error
01	01	DUMPNUM	DS	0CL8	Unique problem identification assigned
01	01		DC	C'PRB'	Problem number prefix
04	04	NUM	DC	C'00000'	Problem number
0C	12	DMPFSCB	DC	F'0'	Address of dump read FSCB
10	16	GRCPARM	DC	F'0'	GETREC parameter list address
14	20	TRNPARM	DC	F'0'	Translate routine PARM2 (data length)
18	24	TRNPARM1	DC	F'0'	Translate routine PARM1 (address)
1C	28	MODDISP	DS	F	Displacement of failure in module
20	32	REALEND	DS	F	Highest address of fixed storage
24	36	EDMRET	DS	F	Return address in DMMEDM
28	40	EDMSAVE	DS	16F	Save area for DMMEDM

113 **** THE PROBLEM REPORT KEYWORD AREA FOLLOWS ****

68	104	KEY	DC	AL2(VMKEYS- KEY),X'0000'	Set initial length
6C	108	COMPLN	DC	X'00160000'	Length of component id keyword
70	112	VMCOMPID	DC	C'VMCOMPID='	Component id keyword
79	121	VCOMP1	DC	C'5749DMK00'	VM/370 component id
82	130	VMENVL	DC	X'000E0000'	Length of environment key
86	134	VMENV	DC	C'VMENVIR=CP'	Environment keyword
90	144	PLCLN	DC	X'000D0000'	Length of PLC keyword
94	148	VM PLC	DC	C'VMPLC='	PLC keyword
9A	154	VM PLC1	DS	CL3	PLC number
9D	157	SCPLN	DC	X'000F0000'	Length of SCP keyword area
A1	161	VM SCPLV	DC	C'VMSCPLV='	SCP keyword
A9	169	VM SCPLV1	DS	CL3	SCP number
AC	172	FAILLN	DC	X'00190000'	Length of failure keyword area

Figure 2-3. VMFDUMP Shared Constant Area (Part 3 of 4)

```
 B0  176       VMFAIL    DC    C'VMFAILURE='  Failure keyword
               VMFAILLP  EQU   *+4            End of VMFAIL if loop or
                                              performance failure
               VMFAILOT  EQU   *+5            End of VMFAIL if other failure
 BA  186       VMFAIL1   DS    CL11           Failure type
               VMKEYS    EQU   *              Length of base keys

     **** OTHER KEYWORDS WHICH MAY OR MAY NOT EE USED FOLLOW****

 C5  197       MODLN     DC    X'00130000'    Length of module name
 C9  201       VMMOD     DC    C'VMMODULE='   Module name keyword
 D2  210       VMMOD1    DS    CL6            Failing module name
 D8  216       DISPLN    DC    X'000F0000'    Displacement key area length
 DC  220       VMDISP    DC    C'VMDISP='     Displacement keyword
 E3  227       VMDISP1   DS    CL4            Displacement
 E7  231       ENTRYLN   DC    X'00140000'    Entry point key area length
 EB  235       VMENTRY   DC    C'VMENTRY='    Entry point keyword
 F3  243       VMENTRY1  DS    CL8            Entry point address
 FB  251                 DC    80C' '         Padded for additional key
14C  332       SUPPORT   DS    F              Supporting data area
150  336       SUPPLN    DC    X'0050'        Supplementary data area length
                                              initially set
152  338       SUPP1     DC    80C' '         Supporting data reserved fcr
                                              the dump fileid
1A2  418       SUPP2     DC    320C' '        User supporting data area

     **** THE PROBLEM REPORT TEXT AREA FOLLOWS: ****

2E2  738       TEXT      DC    X'01E0'        Text area length initially set to 48

     **** THE FIRST 480 BYTES RESERVED FOR THE EXTRACTION PROGRAM ****

2E4  740       TEXTA     DC    80C' '         Data extraction text line a
334  820       TEXT1     DC    80C' '         Data extraction text line 1
384  900       TEXT2     DC    80C' '         Data extraction text line 2
3D4  980       TEXT3     DC    80C' '         Data extraction text line 3
424 1060       TEXT4     DC    80C' '         Data extraction text line 4
474 1140       TEXT5     DC    80C' '         Data extraction text line 5

     **** USER TEXT AREA UP 15 LINES OF 80 BYTE ENTRIES ****
```

Figure 2-3. VMFDUMP Shared Constant Area (Part 4 of 4)

INTSECT -- VMFDUMP AND PROB INTERNAL DATA AREA

```
   ┌──────────────────────────────────────────────────────────────┐
00 |                     INTCPUT                |    INTSER        |
   |────────────────────────────────────────────┘                  |
04 |                                                               |
   |              ┌────────────────────────────────────────────────|
08 |              | INTSDA        | INTSUBC     |    INTTIME       |
   |────────────────────────────────────────────┘                  |
0C |                                                               |
   |───────────────────────────────────────────────────────────────|
10 |  INTSEV      | INTBYPS       | INTPDC      |    INTED         |
   |────────────────────────────────────────────┘                  |
14 |                                                               |
   |                                            ┌──────────────────|
18 |                                            |    INTFILL       |
   |───────────────────────────────────────────────────────────────|
1C |                           INTET                               |
   |              ┌────────────────────────────────────────────────|
20 |              |                INTPNUM                         |
   |───────────────────────────────────────────────────────────────|
24 |                             |           INTTOOL               |
   |─────────────────────────────┘                                 |
   <                                                               >
   <                                            ┌─────────────────>
3C |                                            |    INTPLC        |
   |───────────────────────────────────────────────────────────────|
30 |                             |           INTDUMM               |
   <                                                               >
   <                                                               >
   |───────────────────────────────────────────────────────────────|
58 |                           INTEX1                              |
   <                                                               >
   <                                                               >
   |───────────────────────────────────────────────────────────────|
7C |                           INTX2                               |
   <                                                               >
   <                                                               >
   └───────────────────────────────────────────────────────────────┘
```

Displacement Hex	 Dec	Field Name		Description
00	00	INTSECT	DSECT	
00	00	INTCPUT	DS CL3	CPU type
03	03	INTSER	DS CL6	CPU serial
09	09	INTSDA	DS CL1	Support data available switch
0A	10	INTSUBC	DS CL1	Submitter's code
0B	11	INTTIME	DS CL5	Total time expended
10	16	INTSEV	DS CL1	Problem severity
11	17	INTBYPS	DS CL1	Bypass required switch
12	18	INTPDC	DS CL1	Filler
13	19	INTERRT	DS OCL14	Error date and time
13	19	INTED	DS CL8	Date of error
1B	27	INTFILL	DS CL1	Filler
1C	28	INTET	DS CL5	Time of error
21	33	INTPNUM	DS CL5	System-assigned problem number
26	38	INTTOOL	DS CL25	Tool usage codes
3F	63	INTPLC	DS CL3	PLC level of system
42	66	INTDUMM	DS CL22	Filler
58	88	INTX1	DS CL36	VMFAILURE (internal use only)
7C	124	INTX2	DS CL36	VMENVIR (internal use only)

Figure 2-4. VMFDUMP and PROB Internal Data Area

SYMSECT — SYMPTOM SUMMARY CONTROL RECORD FORMAT

```
   ┌───────────────────────────────────────────────────────┐
00 │                       SYMPNUM                         │
   │───────────────────────────────────────────────────────│
04 │             │      *      │        SYMCREAT           │
   │───────────────────────────┘                           │
08 │                                                       │
   │                           ┌───────────────────────────│
0C │                           │      *      │  SYMLACT    │
   │─────────────────────────────────────────┘             │
10 │                                                       │
   │───────────────────────────────────────────────────────│
14 │      *      │              SYMLFCT                    │
   │─────────────┘                                         │
18 │                                                       │
   │             ┌─────────────────────────────────────────│
1C │             │      *      │        SYMSTAT            │
   │───────────────────────────┘                           │
20 │                                                       │
   │                           ┌───────────────────────────│
24 │                           │      *      │   SYMPTF    │
   │─────────────────────────────────────────┘             │
28 │                                                       │
   │                                                       │
2C │                                                       │
   │                                                       │
30 │                                                       │
   │                                                       │
34 │                                                       │
   │───────────────────────────────────────────────────────│
38 │      *      │   SYMPLC    │             │             │
   │───────────────────────────────────────────────────────│
3C │      *      │   SYMSEV    │      *      │   SYMFAIL   │
   │─────────────────────────────────────────┘             │
40 │                                                       │
   │                                                       │
44 │                                                       │
   │                           ┌───────────────────────────│
48 │                           │      *      │  SYMENVIR   │
   │───────────────────────────────────────────────────────│
4C │             │             │             │  SYMCCNT    │
   └───────────────────────────────────────────────────────┘
```

Figure 2-5. Symptom Summary Control Record Format (Part 1 of 2)

Displacement					
Hex	Dec	Field Name			Description
00	00	SYMSECT	DSECT		
00	00	SYMPNUM	DS	CL5	Problem number
05	05		DS	CL1	Filler
06	06	SYMCREAT	DS	CL8	Creation date for this problem
0E	14		DS	CL1	Filler
0F	15	SYMLACT	DS	CL5	Date of last activity
14	20		DS	CL1	Filler
15	25	SYMLFCT	DS	CL8	Last activity performed
1D	29		DS	CL1	Filler
1E	30	SYMSTAT	DS	CL8	Current status of this problem
26	38		DS	CL1	Filler
27	39	SYMPTF	DS	0CL17	Filename and filetype of PTF for this problem or PNUM or duplicate problem
27	39	SYMPTFFN	DS	CL8	PTF filename
2F	47	SYMPTFDV	DS	CL1	Divider between filename and filetype
30	48	SYMPTFFT	DS	CL8	PTF filetype
38	56		ORG	SYMPTF	
27	39	SYMAPAR1	DS	CL4	Place for APAR
2B	43	SYMAPARX	DS	CL1	Blank divider
2C	44	SYMAPAR2	DS	CL8	APAR number
34	52		ORG	SYMPTF	
27	39	SYMCLOSE	DS	CL17	Closing code if not resolved
38	56		ORG	SYMPTF	
27	39	SYMDUP	DS	0CL17	Field to flag problem as duplicate
27	39	SYMDUP1	DS	CL7	'DUP OF '
2E	46	SYMDUP2	DS	CL5	Problem number of duplicate
33	51	SYMDUPX	DS	CL5	Filler
38	56		DS	CL1	Filler
39	57	SYMPLC	DS	CL3	PLC level of system
3C	60		DS	CL1	Filler
3D	61	SYMSEV	DS	CL1	Severity of this problem
3E	62		DS	CL1	Filler
3F	63	SYMFAIL	DS	CL11	Value of keyword VMFAILURE
4A	74		DS	CL1	Filler
4B	75	SYMENVIR	DS	CL4	Value of keyword VMENVIR
4F	79	SYMCCNT	DS	CL1	Number of keyword symptom records following this header

Figure 2-5. Symptom Summary Control Record Format (Part 2 of 2)

The following CP and CMS data areas are used by IPCS:

CMS

NUCON	CMS low core constant area
FSCB	File system control block

CP

VMBLOK	Virtual machine block
PSA	Prefix storage area
RDEVBLOK	Real device block
RCUBLOK	Real control unit block
RCHBLOK	Real channel block
IOBLOK	I/O control block
BSCBLOK	Binary synchronous control block
VDEVBLOK	Virtual device block
VCUBLOK	Virtual control unit block
VCHBLOK	Virtual channel block
DMPINREC	Dump file information record
DMPKYREC	Dump file key record
DMBTBREC	Dump file symbol table record

These data areas are described in the *VM/370 Data Areas and Control Blocks Logic*.

Diagnostic Aids

Figure 2-6 is an alphabetical list of all the messages issued by IPCS. The nearest label and the associated method of operation diagram are identified.

Message Code	Label	Diagram	Message Text
DMMCPA805I	CPAEND	2-8	xxxxxx ABEND CODE NOT RECOGNIZED BY DATA EXTRACTION
DMMDSC700I	GETDUMP	2-1	TYPE HELP OR ENTER
DMMDSC701R	ENTER	2-1	ENTER DUMP NUMBER AND MODE
DMMDSC719I	STATERR	2-1	ERROR IN FSSTATE
DMMDSC720I	TWOMAPS	2-1	LOAD MAP ALREADY PRESENT
DMMDSC721I	FULLMSG	2-1	&NAME TABLE IS FULL
DMMDSC722I	ILLEGAMP	2-1	INVALID ENTRY INTO THE &NAME TABLE
DMMDSC723I	NOWRITE	2-1	THE DUMP IS NOT ON THE A-DISK
DMMEDM100S	PRBQUITR	2-5	ERROR 'nnn' READING FILE 'SUMMARY RECORD A'
DMMEDM200S	PRBWRTER	2-5	ERROR 'nnn' WRITING FILE 'SUMMARY RECORD A'
DMMEDM400S	PRBQUITC	2-5	ERROR 'nnn' CLOSING FILE 'SUMMARY RECORD A'
DMMEDM850I	LOOP	2-5	UNABLE TO READ DUMP FROM READER
DMMEDM852I	ERRWRT	2-5	FATAL I/O ERROR WRITING DUMP
DMMEDM853I	NODMP	2-5	NO DUMP FILES EXIST
DMMEDM860I	QUIT	2-5	FATAL I/O ERROR READING DUMP
DMMEDM861E	ERROR3	2-5	DUMP FILE FILENAME NOT FOUND
DMMEDM863E	ERRFND	2-5	INVALID PARAMETER - XXXXXXXX PAGE REFERENCED NOT AVAILABLE
DMMFED702I	CBADIN	2-1	NON-HEX CHARACTER IN COUNT - RETRY
DMMFED703I	BADIN	2-1	NON-HEX CHARACTER IN ADDRESS - RETRY
DMMFEX704I	BADIN	2-1	NON-HEX CHARACTER IN ADDRESS - RETRY
DMMGET100S	RDERR	x	ERROR 'nnn' READING FILE 'fileid'
DMMGET708I	MSGITEND	x	PAGE 'page' NOT FOUND IN THE DUMP
DMMGRC100S	RDERROR	x	ERROR 'nnn' READING FILE 'PRBnnnnn DUMP A1'

Figure 2-6. Interactive Problem Control System Messages (Part 1 of 3)

Message Code	Label	Diagram	Message Text
DMMGRC809S	MSGITEND		REQUESTED ADDRESS NOT IN DUMP
DMMHEX714I	BADIN	2-1	NON-HEX CHARACTER IN INPUT - RETRY
DMMINI100S	EXTERR	2-5	ERROR 'nnn' READING FILE 'PRBnnnnn DUMP A1'
DMMINI400S	EXTERRC	2-5	ERROR 'nnn' CLOSING FILE 'PRBnnnnn DUMP A1'
DMMINI800S	WRTMSG	2-5	DATA EXTRACTION FAILURE
DMMINI803S	ERR202	2-5	ERROR 'nnn' ATTEMPTING LOADMOD FOR VMFDUMP2
DMMIOB712I	WRITERR	2-1	DEVICE 'cuu' NOT FOUND
DMMIOB713I	NOUSER	2-1	USER 'userid' VMBLOK NOT FOUND
DMMLOC715I	NONHEXST	2-1	NON-HEX CHARACTER IN STRING
DMMLOC716I	MOREMSG	2-1	STRING 'string' NOT FOUND
DMMLOC716I	NOTOFF	2-1	STRING 'string' NOT FOUND BEFORE END OF DUMP
DMMLOC717I	NOPARM	2-1	INVALID FORM OF LOCATE COMMAND
DMMMAP200S	WRTERR	2-6	ERROR 'nnn' WRITING FILE 'PRBnnnnn DUMP A1'
DMMMAP801I	MAPERROR	2-6	FILE 'NUC MAP' IS NOT VALID FOR THIS DUMP
DMMMAP802I	CONTMSG	2-6	PROCEEDING..
DMMMAP806R	MAPNAME	2-6	ENTER fn ft fm OF THE NUCLEUS LOAD MAP
DMMMAP807I	STATERR	2-6	UNABLE TO LOCATE 'fileid'
DMMMAP808I	MAPERROR	2-6	NUCLEUS MAP INVALID 'fileid'
DMKMAP810S	READERR	2-6	ERROR 'nnn' READING FILE 'fileid'
DMMMOD100I	RDERR	2-1	ERROR 'nnn' READING FILE 'fileid'
DMMMOD705E	EOTAB	2-1	ERROR IN ITEM TABLE
DMMMOD706I	MODNF	2-1	'entry name' NOT FOUND IN THE LOAD MAP

Figure 2-6. Interactive Problem Control System Messages (Part 2 of 3)

Message Code	Label	Diagram	Message Text
DMMMOD707I	GETREAL	2-1	'module' 'page' PAGE NOT VALID
DMMMOD718I	LMERR	2-1	THIS DUMP HAS NO LOAD MAP - SEE MAP SUBCOMMAND
DMMPRM200S	PRBWRTER	2-5	ERROR 'nnn' WRITING FILE 'SUMMARY RECORD A1'
DMMPRM804I	START	2-5	ERROR IN DATA EXTRACTION
DMMPRO100S	SUMERRR2	2-3	ERROR 'nnn' READING FILE 'fileid'
DMMPRO200S	SUMERRW	2-3	ERROR 'nnn' WRITING FILE 'fileid'
DMMREG100I	RDERR	2-1	ERROR 'nnn' READING FILE 'fileid'
DMMSCR709I	EYECATCH	2-1	NO VALID SCROLL ADDRESS
DMMSEA100S	RDERR	2-1	ERROR 'nnn' READING FILE 'SYMPTOM SUMMARY A1'
DMMSTA100S	RDERR4	2-4	ERROR 'nnn' READING FILE 'fileid'
DMMSTA200S	WRTERR	2-4	ERROR 'nnn' WRITING FILE 'fileid'
DMMSTA601S	CK2CONT	2-4	OPERAND NOT RECOGNIZED, STATALL ASSUMED
DMMSUM100S	RDERR	2-2	ERROR 'nnn' READING FILE 'SYMPTOM SUMMARY A1'
DMMSUM200S	ERRWRT	2-2	ERROR 'nnn' WRITING FILE 'fileid'
DMMSUM501S	PARMERR	2-2	INVALID PARM 'parm' PASSED TO SUMMARY UPDATE PROGRAM
DMMSUM502S	RETCOD4	2-2	PROBLEM 'PRBnnnnn' NOT FOUND IN SYMPTOM SUMMARY
DMMTRC710I	CBADIN	2-1	NON-NUMERIC COUNT CHARACTER - RETRY
DMMVMB711I	NBA	2-1	LOOP IN VMBLOK CHAIN
DMMWRT200S	FSWRITE	2-3	ERROR 'nnn' WRITING FILE 'fileid'

Figure 2-6. Interactive Problem Control System Messages (Part 3 of 3)

Chapter 3. The Format Service Program

Introduction

The Format/Allocate service routine is a standalone prcgram which:

- Formats all or part of a DASD device
- Allocates DASD space
- Creates volume labels

for IBM 2314, 2319, 3330, 3340, 3350 series, and 2305 series direct access storage devices.

Operands entered from the IPL device and/or a 1052 console ccntrol the execution of the Format program.

FORMAT OPERATION

The Format program writes 4096-byte (one page) records on all the specified cylinders. The records just written are then read tc verify the disk surface. Any records not passing the read-after-write check are counted. When the format operation is complete, a summary of the addresses of the unusable pages is written on the console.

The first three records of cylinder 0 contain special system data including the volume label. If the format operation includes cylinder 0 any existing volume label is read first and if an OS Format 4 label is present, the information in the label concerning alternate track assignments is carried forward to the new label. Then the new volume label is written on the DASD device.

If cylinder 0 is not to be formatted, label checking is performed.

If unrecoverable DASD errors occur during the formatting operation, the format function is canceled, the message

```
DMKFMT735E  FATAL DASD I/O ERROR
```

is issued, and the next control statement is read.

LAEEL-ONLY OPERATICN

In a label-only operaticn, a new volume latel is written on cylinder 0, track 0, record 3 of the DASD device. No label checking is done tefore the new label is written. The device must already be formatted before a latel operation can be performed.

ALLOCATION OPERATICN

In an allocation operaticn, disk space is assigned on the specified device in units of one cylinder. This disk space may be used as:

- Temporary space (TEMP)
- Permanent space (PERM)
- Cirectory space (DRCT)
- Temporary user space (TCSK)

The input parameters provide the information needed to update the allocaticn table. When the END allocation statement is processed:

- The allocation table is written in the byte allocation map on cylinder 0, track 0, record 4 of the DASC device.
- The results of the allocaticn cperation are displayed at the console.

The CASD device must already be formatted before an allocaticn operation can be performed.

EXECUTING THE FORMAT PROGRAM

The sequence for executing the Format program is:

1. Ready the DASD device.
2. Ready the reader. The reader must contain the Format/Allccate program and may also contain ccntrol cards fcr the program.

3. IPL the reader.

4. If a console is not located at either address 009 or 01F, signal attention from the console so the Format program can establish the address of the console.

5. The program title is printed.

6. When there are no control cards in the reader, the program requests control statements by sending prompting messages to the console.

7. When control cards are in the reader, they are processed. The prompter messages are displayed with the response field updated from the control statements already entered through the card reader. The program requests additional input, which can be entered via the reader or console.

8. The program issues messages indicating the start or end of an operation.

9. An operation in progress may be canceled by signaling attention from the console. Execution resumes with the next operation.

10. The Format/Allocate program cancels an operation if a an unrecoverable DASD I/O error occurs. A message indicating the cause of the error is displayed.

Method of Operation

This section describes the execution of the disk format program and shows the processing associated with:

- Formatting DASD space.
- Allocating DASD space.
- Writing a volume label.

Figure 3-1 shows the relaticnship cf the diagrams.

Diagram 3-1 describes the major functions of the Format/Allccate program.

Diagram 3-2 describes the fcrmat function of the Format/Allocate program.

Diagram 3-3 describes the allccate function of the Format/Allccate prcgram.

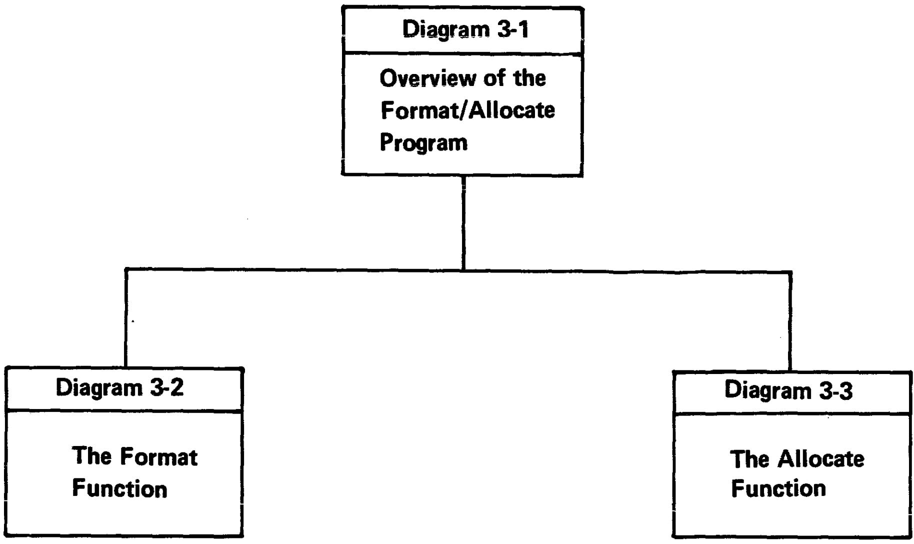
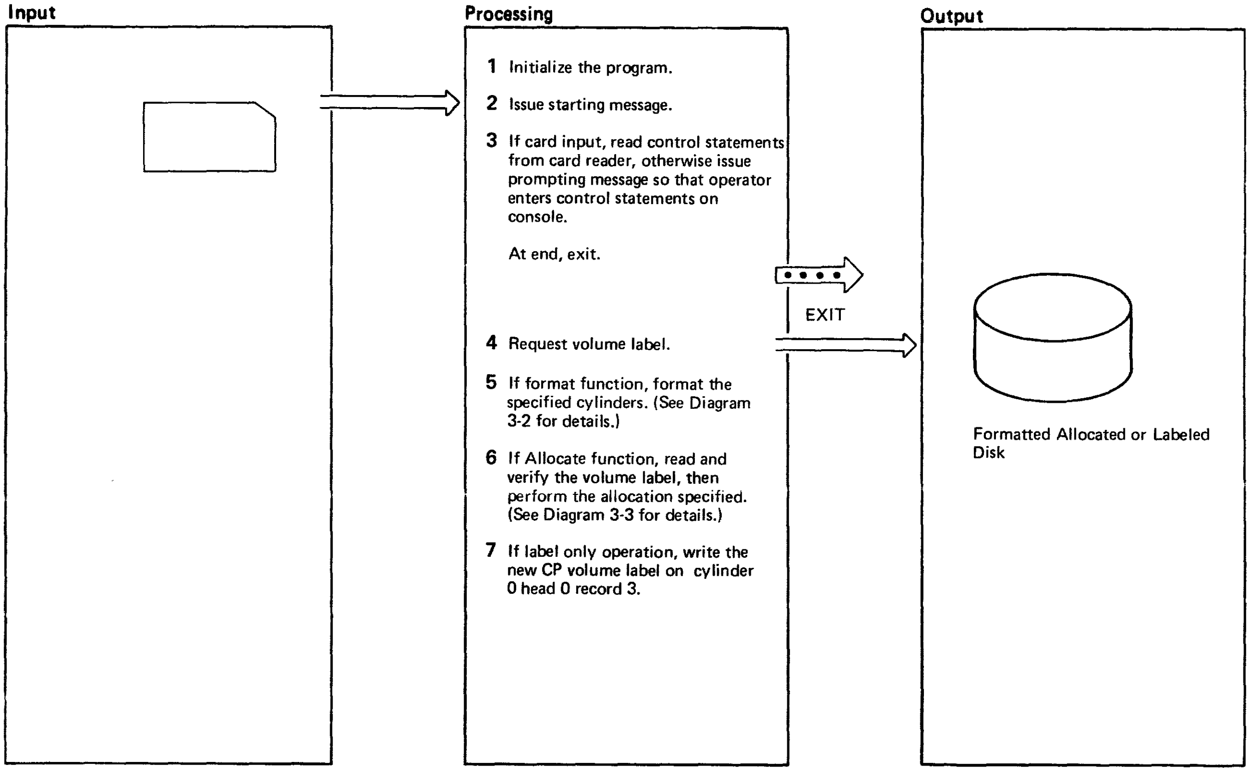


Figure 3-1. Key to the Format/Allocate Program Method of Operation Diagrams



1 Initialize the program.

2 Issue starting message.

3 If card input, read control statements from card reader, otherwise issue prompting message so that operator enters control statements on console.

At end, exit.

4 Request volume label.

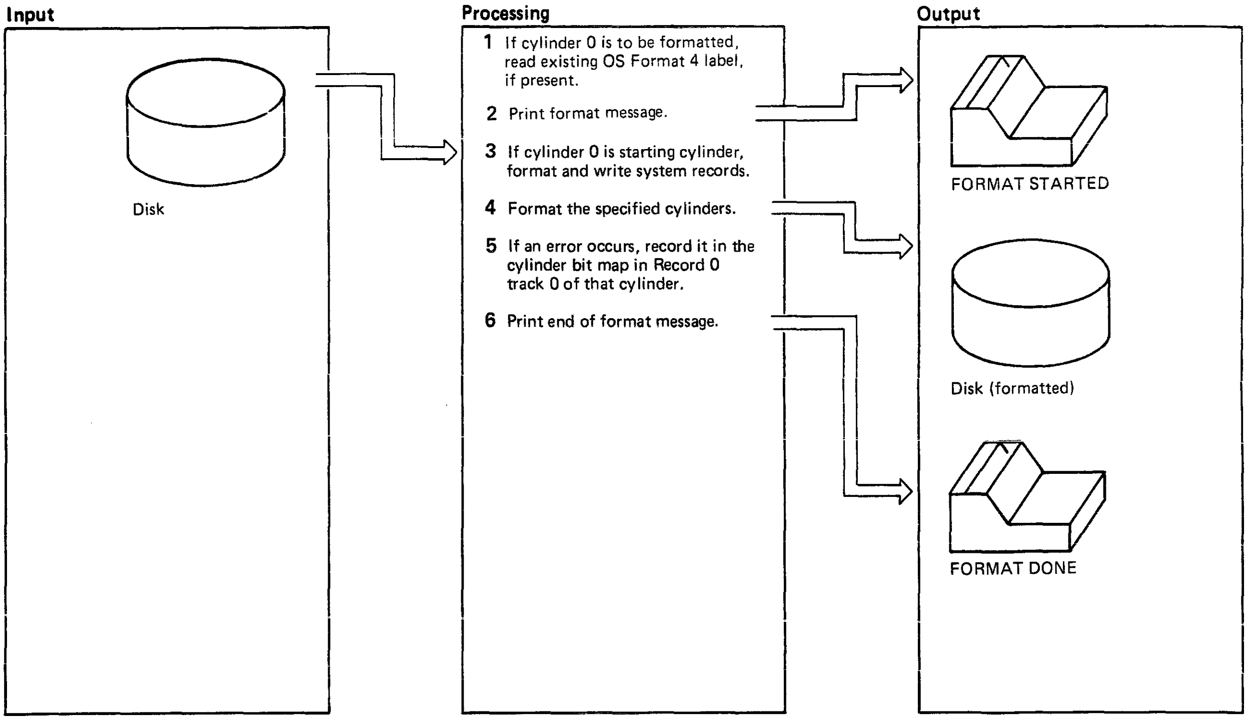
5 If format function, format the specified cylinders. (See Diagram 3-2 for details.)

6 If Allocate function, read and verify the volume label, then perform the allocation specified. (See Diagram 3-3 for details.)

7 If label only operation, write the new CP volume label on cylinder 0 head 0 record 3.

Notes	Module	Label	Ref
1 DMKFMT sets up registers 15, 11, and 12 as base registers, gets the IPL device address from the I/O old PSW, and stores it in IPLDEV. Next, DMKFMT locates the console by testing 009 and 01F. If neither of these devices is available, it enters the wait state until an attention interruption is received from the console.	DMKFMT	DMKFMT	
2 The program title VM/370 FORMAT/ALLOCATE PROGRAM VERSION n.n is displayed at the console.	DMKFMT	STMSG	
3 If the switch (CDSW2) contains I'FF', the reader enters the wait state until an I/O interrupt occurs. The CONSINT routine reads the control statements and the VALIDATE routine checks that they are valid.	DMKFMT	GETCARD CONSINT VALIDATE	
The prompter messages are issued. If the control statements are entered through the card reader, the prompter messages include the response that was already specified in cards.		SELECT	
The message ENTER FORMAT OR ALLOCATE: prompts the operator. If the operator correctly enters FORMAT (F) or ALLOCATE (A), one of the following messages FORMAT FUNCTION SELECTED ALLOCATE FUNCTION SELECTED appears on the console. Otherwise, the prompter message is reissued. Then, the message		DEVICEAD	
ENTER DEVICE ADDRESS (ccu): prompts the entering of the device address. If the device address entered is valid, the device type is requested.		DEVICEAD	
ENTER DEVICE TYPE: The high cylinder address, highest record, and device type are initialized depending on the device type entered. If the device address entered is not available, the error message DMKFMT730E DEV xxx NOT OPERATIONAL is issued and the request for a device is repeated.		DEVTYPE	
4 The message ENTER DEVICE LABEL: is displayed.	DMKFMT	LAB	
5 If the function being performed by the Format/Allocate program is the format operation, then, if cylinder 0 is to be formatted, DMKFMT branches to FMT; otherwise, it branches to REGFORM1.	DMKFMT	LAB	
6 The volume label is read and verified by the LBLREC CCW string, then DMKFMT branches to the ALLOCATE routine.	DMKFMT	LAB	
7 The CP volume label is written by the LABWRITE CCW string. Processing continues by reading the next control statement (see Step 3).	DMKFMT	LABONLY	

Diagram 3-1. Overview of the Format/Allocate Program



1 If cylinder 0 is to be formatted, read existing OS Format 4 label, if present.

2 Print format message.

3 If cylinder 0 is starting cylinder, format and write system records.

4 Format the specified cylinders.

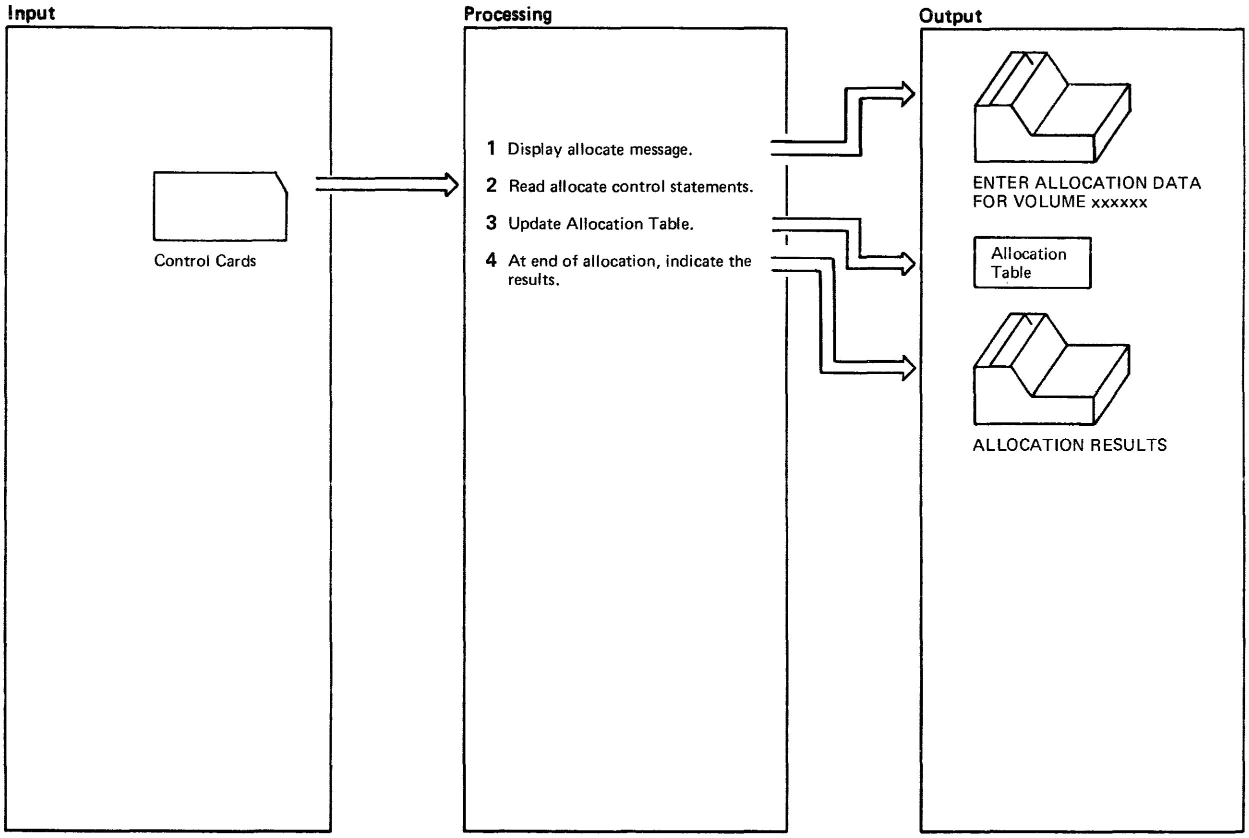
5 If an error occurs, record it in the cylinder bit map in Record 0 track 0 of that cylinder.

6 Print end of format message.

Notes	Module	Label	Ref
1 If cylinder 0 is to be formatted, any existing OS Format 4 label is read to preserve (for IBCDASDI) the CCHH address of the next unassigned alternate track and also the count of the remaining unassigned alternates. This data will be put in the new OS Format 4 label on track 0.	DMKFMT	FMT	
2 DMKFMT branches and links to the message writing (WMSG) routine to display FORMAT STARTED Then it updates the I/O new PSW so that the IOINT routine executes when an I/O interrupt occurs.	DMKFMT	REGFORM1	
3 If cylinder 0 is the starting cylinder, the FORMAT program formats cylinder 0 by setting up the CCWs appropriate to the device type and then branching to the STIO routine to perform the I/O operation. Once cylinder 0 is formatted, system records are written on it. The branch to the CHECK0 routine is set to NOP so that CHECK0 is executed only once. The records written on cylinder 0 are Record Description 0 Page bit map 1 IPL record 2 Checkpoint record 3 Vol1 label 4 Allocation bit map 5 Format 4 label 6 Format 5 label F3 Page size filler F4 Filler record for 2314/2319	DMKFMT	STORE CHECK0	

Notes	Module	Label	Ref
4 The appropriate device type CCWs are set up by the Format program. Page size records are written and verified by the STIO routine. Control returns to the RESUMP routine if no error occurs. The RESUMP routine updates the record numbers and the STIO routine again writes and verifies the record. This loop continues until the last cylinder specified is completely formatted.	DMKFMT	STORE STIO RESUMP	
5 If an error occurs in the STIO routine, control is transferred to the IOINT routine. The error is retried up to 9 times before the message DMKFMT736E IO ERROR xxx CCHHR= SENSE= is displayed. The Page bit map is updated to indicate a bad surface. The errors that cause the Format function to terminate are: seek error error in writing or reading the home address error writing or reading record 0 error setting file mask error in reading count, key, data The message DMKFMT735E FATAL DASD I/O ERROR is displayed and control returns to the GETCARD routine.	DMKFMT	IOINT READER06	
6 DMKFMT displays the message FORMAT DONE to indicate that the specified cylinders are formatted, and then summarizes the errors with the message xxx PAGE RECORDS FLAGGED	DMKFMT	CLEANUP	

Diagram 3-2. The Format Function



1 Display allocate message.

2 Read allocate control statements.

3 Update Allocation Table.

4 At end of allocation, indicate the results.

Notes	Module	Label	Ref
1 The messages ENTER ALLOCATION DATA FOR VOLUME xxxxxx type cyl cyl .... are displayed.	DMKFMT	ALLOCATE	
2 If the Allocate control statements are entered via a card reader, the switch (CDSW2) contains X'FF'. Control is transferred to the GETCARD routine which reads the cards. The CONSINT and VALIDATE routines verify the control statements and allocate processing resumes at the label REREAD. There is a branch and link to the RMSG routine to read from the console. The console read is not performed in this case because CDSW2 is X'FF'.	DMKFMT	GETCARD CONSINT VALIDATE REREAD	
If the allocate control statements are entered via the console, the switch (CDSW2) contains X'00'. The control statements are read from the console by branching and linking to the RMSG routine.		REREAD RMSG	
3 The address of the cylinder byte map is loaded into register 9. The total number of cylinders specified is loaded into register 8. The cylinder byte map is updated for each of the specified cylinders according to the type indicated in the control statement. Control Statement — Indication in Cylinder Byte Map TEMP X'00' PERM X'01' TDSK X'02' DRCT X'04' The map is printed after the END statement is processed.	DMKFMT	AOKALL INDIC	
4 The message ALLOCATION RESULTS followed by the type corresponding to the allocated cylinders is displayed. Finally, the message DEVICE xxx VOLUME xxxxxx ALLOCATION ENDED is displayed.	DMKFMT	FINI	

Diagram 3-3. The Allccate Function

Program Organization

DMKFMT

A standalone program that formats, allocates, and labels all (or part) of 2314, 2319, 3330, 3340, 3350 series, and 2305 series direct access storage devices for VM/370 use.

Entry Point
DMKFMT

Routines Called
None

Register	Usage
R0-10:	Scratch
R11:	3rd base register
R12:	2nd base register
R14:	Scratch
R14:	Linkage register
R15:	1st base register

Directory

Figure 3-2 is an alphabetical list of the major labels in the Format/Allocate program. The associated method of operation diagram and a brief description of the function performed at the point in the program indicated by each label are included in the list.

Label	Diagram	Description
ALLOCATE	3-3	Performs the allocate function of the Format program.
ALTTRACK		Performs alternate track recovery for 3340/3344.
AOKALL	3-3	Locates the cylinder byte map.
CHECK0	3-2	Writes system records on cylinder 0.
CLEANUP	3-2	Summarizes the errors encountered while formatting the disk.
CONSINT	3-1 3-3	Processes console interrupts.
DEVICEAD	3-1	Displays the prompter message requesting the device address.
DEVTYPE	3-1	Displays the prompter message requesting the device type.
DMKFMT	3-1	Initializes the Format program.
ERRECOV		Performs DASD error recovery.
FATAL		Displays the termination message and reads the next control statement.
FINI	3-3	Displays the cylinders just allocated with the type of allocation.
FMT	3-2	Initializes cylinder 0 for formatting by first reading any existing OS Format 4 label.
FORMAL		Displays the starting cylinder or label message.
GETCARD	3-1 3-3	The main control routine. It reads control statements from the reader or transfers control to the SELECT routine to issue prompter messages.
GRAPHID		Handles input and output operations for display terminals.
INDIC	3-3	Updates the cylinder byte map to reflect the type of allocation for each cylinder.
IOINT	3-2	Handles I/O interrupts and retries errors.
LAB	3-1	Displays the prompter message requesting the device label.
LABELRD	3-1	Reads and verifies the volume label.
LABONLY	3-1	Rewrites the volume label (record 3) and nothing else.
MCRTN		Processes machine checks.
NEXT		Displays end of cylinder message.
ORCHK		Processes program checks.
PRINTALL		Displays the allocation table on the terminal.
READER06	3-2	Updates the page bit map to indicate a bad surface.
REGFORM1	3-2	Initializes the format function when cylinder 0 is not included.
REREAD	3-3	Reads control statements from the console for the allocate function.
RESUMP	3-2	Updates the record number during the format operation.
RMSG	3-3	Reads from the typewriter terminals.
SELECT	3-1	Prompts the operator to enter the appropriate control statement.
SENSIT		Gets sense information.
SENSIT2		Displays the sense information.
STIO	3-2	Writes and verifies page size records during format operation.
STMSG	3-1	Displays the program title.
STORE	3-2	Sets up CCW string to format cylinder 0.
VALIDATE	3-1 3-3	Checks control statements entered through a card reader for accuracy.
WMSG		Displays messages on the terminal.
XBIN		Converts hexadecimal numbers to binary.

Figure 3-2. The Format/Allocate Program Label Directory

Data Areas

This section contains descriptions of the DASD record formats and the layout of these DASD records for:

- 2305 Models 1 and 2
- 2314/2319 devices
- 3330 series
- 3340 series
- 3350 series

Record 0

Record 0 is the standard 8-byte data area, set to 0 except for cylinder 0. On cylinder 0, pages are flagged as follows:

For 2305, 2314/2319, 3330, 3340, and 3350 in compatibility mode:

E0	00	00	00	00	00	00	00

For 3350 in native mode only:

F0	00	00	00	00	00	00	00

Figure 3-3. Record 0 Format

Record 1

24 bytes - track 0 cylinder 0

IPL record - puts system into wait state if storage device is loaded (via IPL functicn).

```
00 02 00 00 00 00 00 00 03 00 00 00 20 00 00 C0 00 00 00 00 00 00 00 00
```

Figure 3-4. Record 1 Format

Record 2

4096 bytes - track 0 cylinder 0

Checkpoint record - this is the Checkpoint program load at VM/370 IPL time to retrieve and save control information for a warm start.

Figure 3-5. Record 2 Format

Record 3 ccnsists of a 4-byte key and an 80-byte record.

Key

key
0 3

Data

Key	Label	F0	Pointer to VTOC	00	40	00	CP370	40	Pointer to user Directory	40
0	4	10	11	16	21	41	46	51	52	56 79

Figure 3-6. Record 3 Format

Record 4

1024 bytes - track 0 cylinder 0

Record 4 is an allocation byte map used to identify cylinder usage. Each byte ccrresponds to one cylinder; the value of the byte indicates the type cf usage for the cylinder.

Value	Usage
00	temporary
01	permanent
02	T-disk
04	directory

Example

```
| 00 00 01 00 04 02 00 . . . . FF 00 . . . . . . . 00 |
                                ▲
                                |

               Cylinder beyond the last cylinder
               that can be allocated.  This point varies
               depending on the device.
```

Figure 3-7. Record 4 Format

Record 5

44 bytes key - track 0 cylinder 0
96 bytes data area

Format 4 DSCB type label - used to be compatible with other systems.

```
| 04---->04 |    | FORMAT 4 LABEL |
44-byte Key       96-byte Data Area
```

Figure 3-8. Record 5 Format

Record 6

44 bytes key - track 0 cylinder 0
96 bytes data area

Format 5 DSCB type label - used to be compatible with other systems.

```
 ┌────────────────┐    ┌────────────────┐
 | 05 05 05 05 00 |    | FORMAT 5 LABEL |
 └────────────────┘    └────────────────┘
    44 Byte Key         96 Byte Data Area
```

Figure 3-9. Record 6 Format

Record F3

4096 bytes - 1 page, track 0 or track 1 (2314/2319 devices only).

F3 Record is reserved for IBM use, and is referred to as a filler record.

Figure 3-10. Record F3

Record F4

1624 bytes - track 1 (2314/2319 devices only).

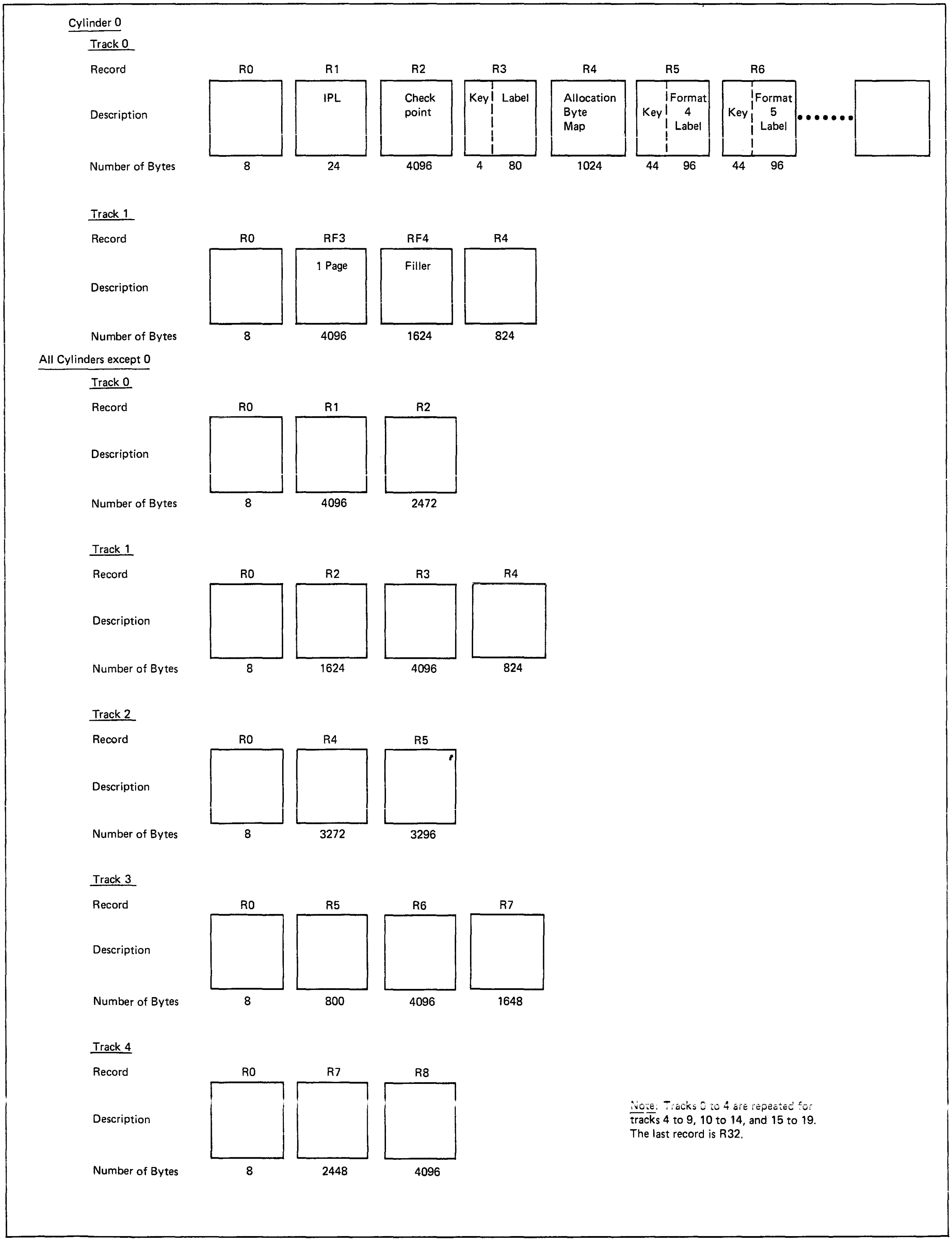
Record F4 is used only on 2314 and 2319 to align Record 4 on the track.

Figure 3-11. Record F4

Record 4

824 bytes - cylinder 0, track 1 (2314/2319 devices only).
The first segment of Record 4 is used for paging.

Figure 3-12. Record 4



Cylinder 0

Track 0

Record	R0	R1	R2	R3		R4	R5		R6		
Description		IPL	Check point	Key	Label	Allocation Byte Map	Key	Format 4 Label	Key	Format 5 Label	•••••••
Number of Bytes	8	24	4096	4	80	1024	44	96	44	96	

Track 1

Record	R0	RF3	RF4	R4
Description		1 Page	Filler	
Number of Bytes	8	4096	1624	824

All Cylinders except 0

Track 0

Record	R0	R1	R2
Description			
Number of Bytes	8	4096	2472

Track 1

Record	R0	R2	R3	R4
Description				
Number of Bytes	8	1624	4096	824

Track 2

Record	R0	R4	R5
Description			
Number of Bytes	8	3272	3296

Track 3

Record	R0	R5	R6	R7
Description				
Number of Bytes	8	800	4096	1648

Track 4

Record	R0	R7	R8
Description			
Number of Bytes	8	2448	4096

Note: Tracks 0 to 4 are repeated for tracks 4 to 9, 10 to 14, and 15 to 19. The last record is R32.

Figure 3-13. 2314/2319 Record Layout

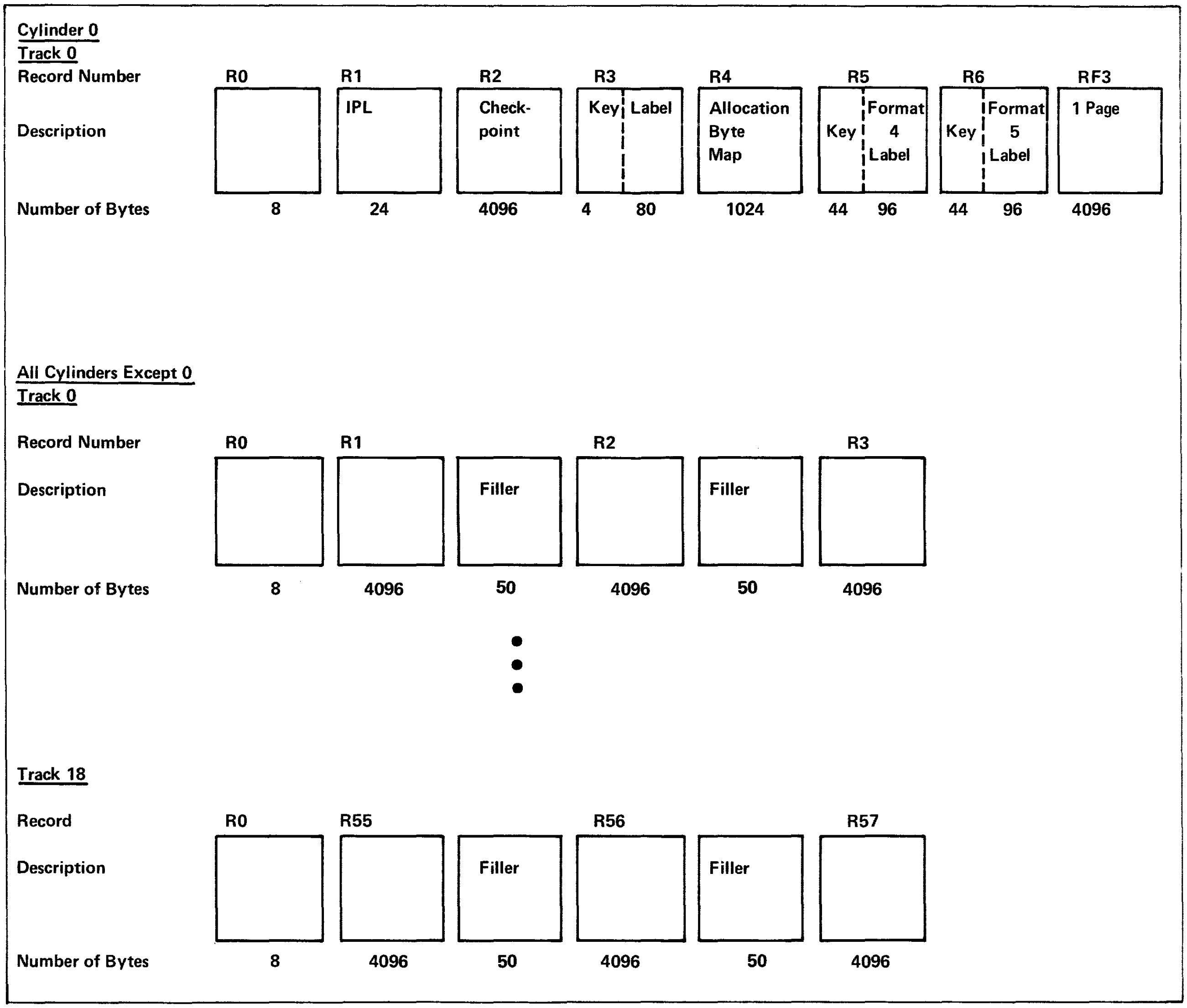


Figure 3-14. 3330 Series Record Layout

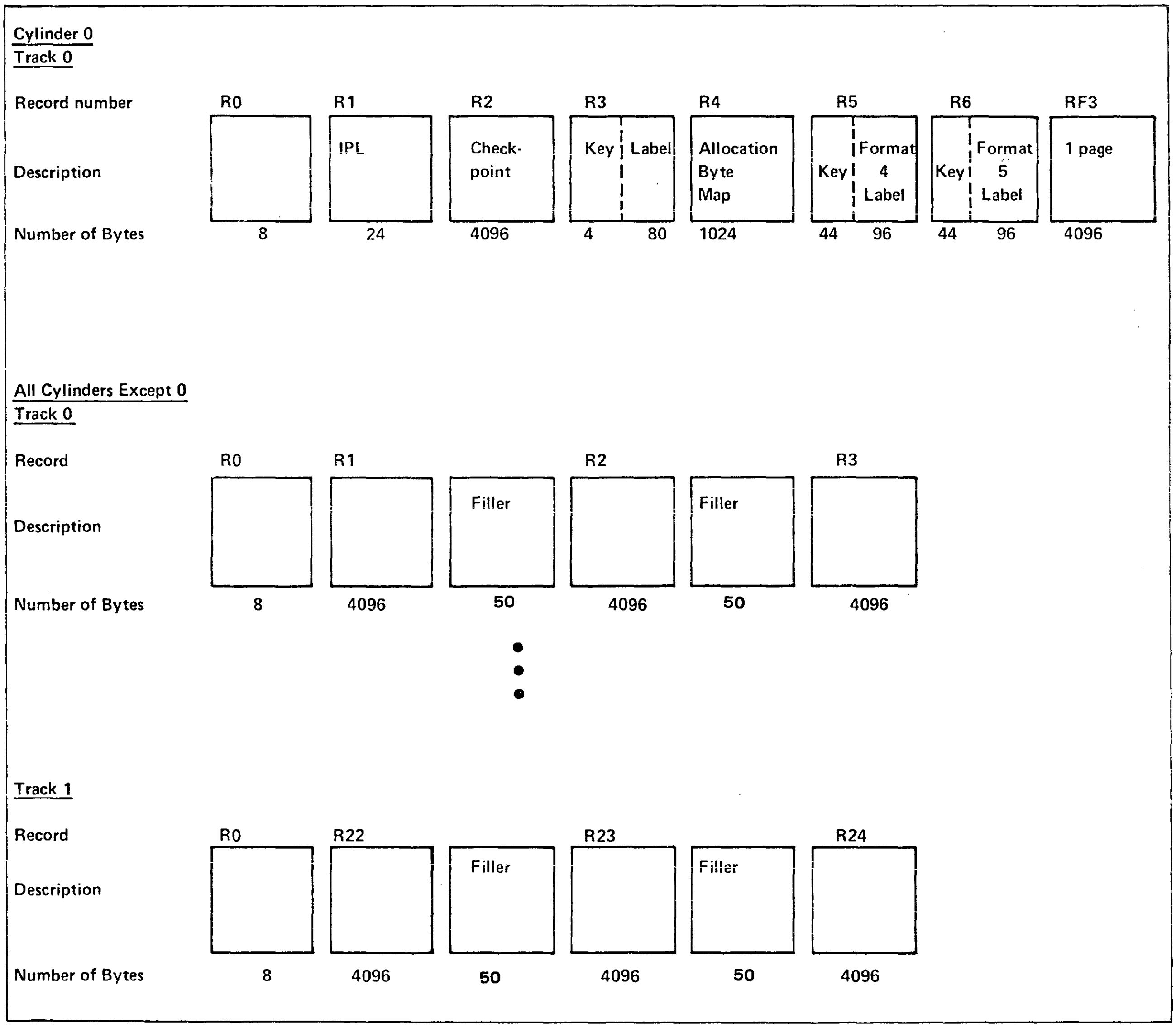


Figure 3-15. 2305 Models 1 and 2 Record Layout

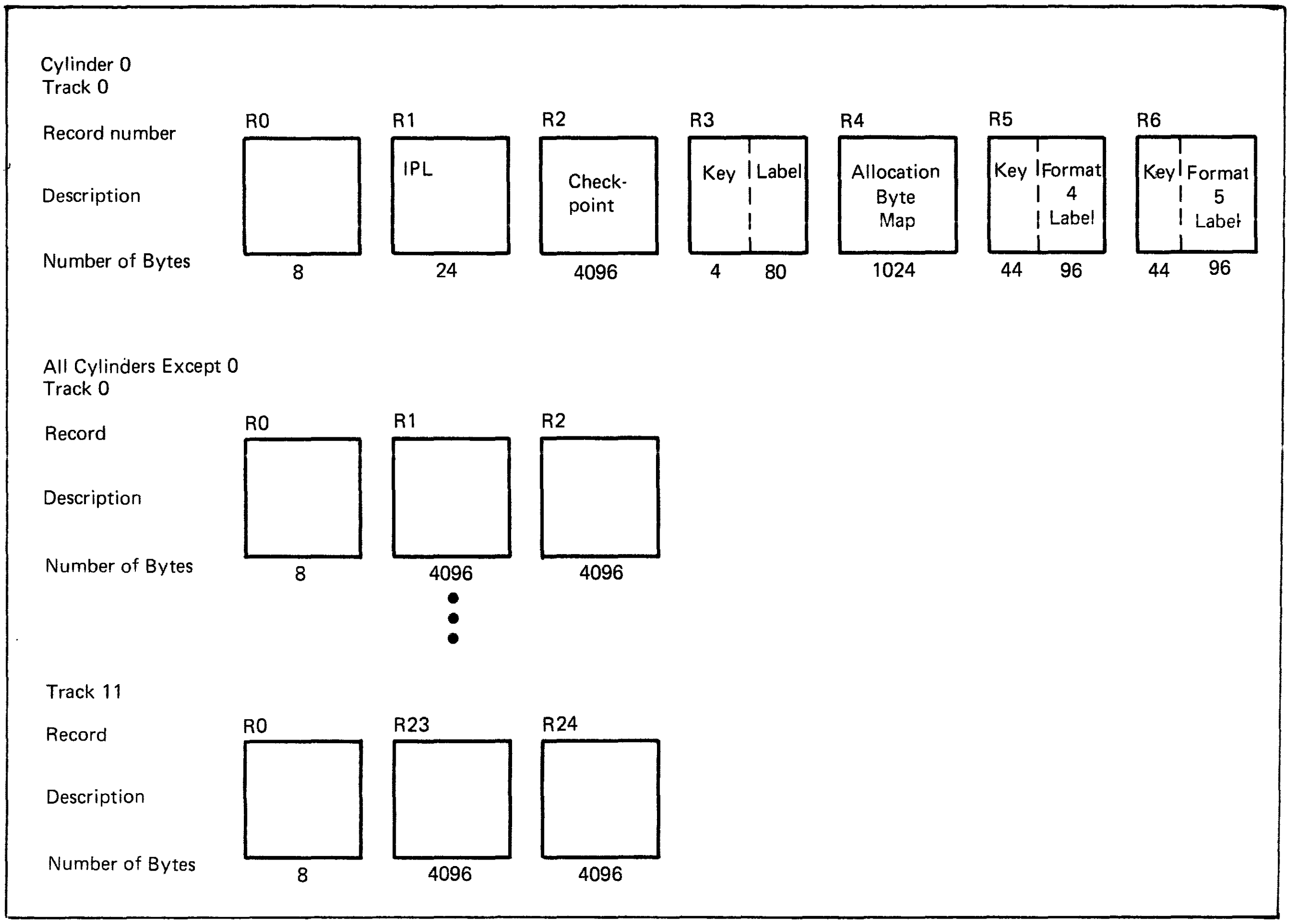


Figure 3-16. 3340 Record Layout

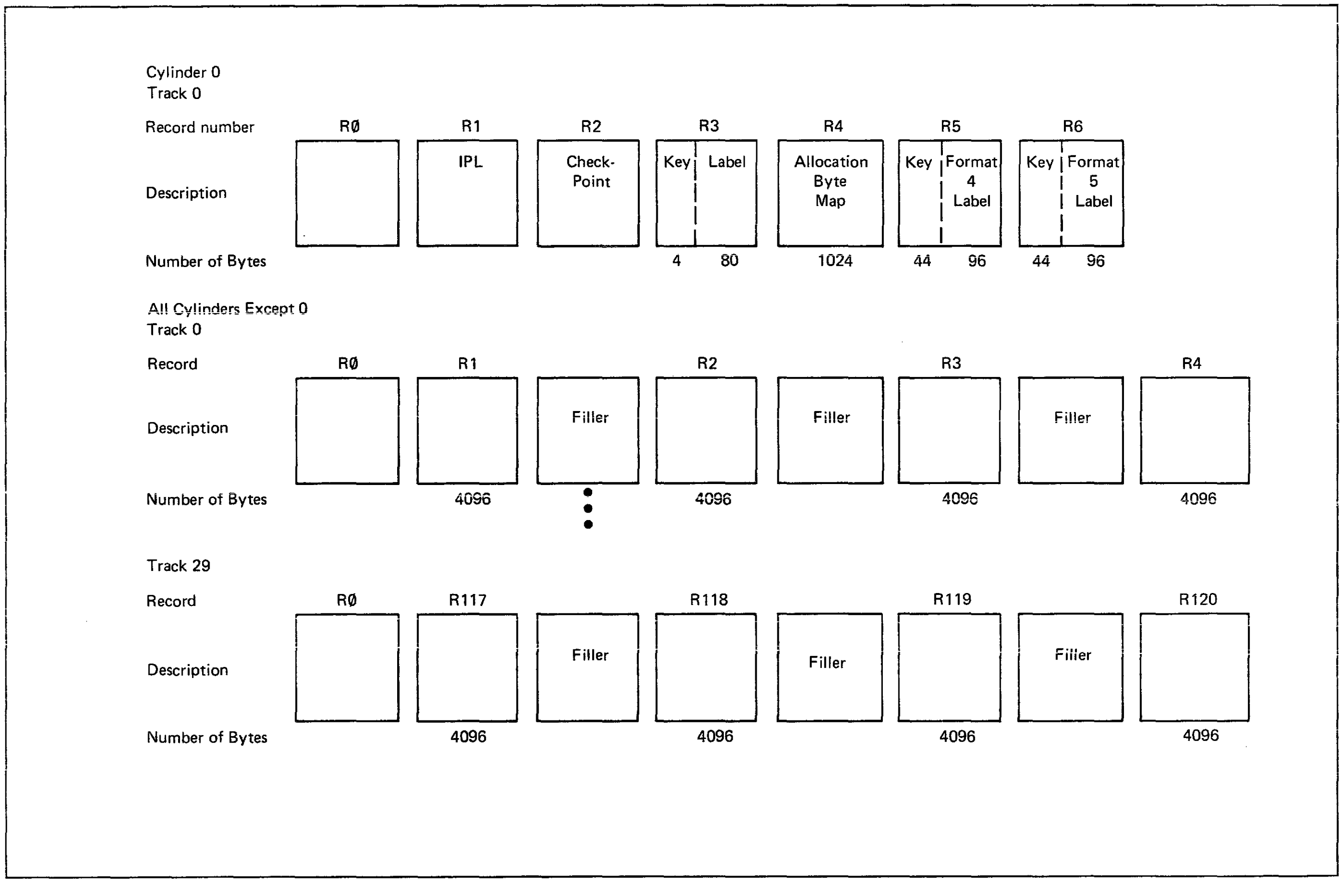


Figure 3-17. 3350 Record Layout

Diagnostic Aids

Figure 3-18 is a list of the messages issued by the Format/Allocate program. The label of the message, the label of the routine issuing the message and the associated method of operaticn diagram are included in the list.

Message Code	Label of Message	Issuing Routine	Diagram	Message Text
DMKFMT730E	WR1	DEVTYPE	3-1	DEV xxx NOT OPERATICNAL OR NCT REACY.
DMKFMT732E	MCMSG	MCRTN		MACHINE CHECK RUN SEREP AND SAVE OUTPUT FOR CE
DMKFMT733E	WRONG	LABELBAD		VOLIC READ IS xxxxxx NCT xxxxxx
DMKFMT734E	TYPERR	VALIDATE ALLOCATE		TYPE OR CYL INVALID
DMKFMT735E	FATLMSG	FATAL	3-2	FATAL DASC IO ERROR. CSW=xxxxxxxxxxxxxxxx
DMKFMT736E	IOERR	DEVICEAD SENSIT2	3-2	IO ERROR xxx CCHHR = 0000000000 SENSE = xxxxxxxxxxxx
DMKFMT737E	BAD	BADINPUT		INVALID OPERAND
DMKFMT738A	IPLERROR	DEVICEAD		DEV xxx INTERVENTION REQUIRED
DMKFMT739E	MSGATRK	ALTTRACK		FLAGGED PRIMARY TRACK HAS NC ALTERNATE ASSIGNED, IO ERRCR FOLLOWS.
DMKFMT740E	MSG35MB	DEVTYPE	3-1	PACK MOUNTED IS 3340-35, NCT 3340-70. MOUNT ANOTHER OR RESPECIFY.
DMKFMT756E	PCMSG	PRCHK ERRORMSG		PROGRAM CHECK PSW= xxxxxxxxxxxxxxxx
	TITLE	STMSG	3-1	VM/370 FORMAT/ALLCCATE PRCGRAM RELEASE n
	FORA	SELECT	3-1	ENTER FORMAT OR ALLCCATE:
	FMTMSG	SELECT	3-1	FORMAT FUNCTION SELECTED
	ALLOCMSG	SELECT	3-1	ALLOCATE FUNCTION SELECTED
	ADDRESS	DEVICEAD	3-1	ENTER DEVICE ADDRESS (CCU):
	TYPMSG	DEVTYPE	3-1	ENTER DEVICE TYPE:
	DATAMSG	ALLOCATE	3-3	ENTER ALLOCATION DATA FOR VCLUME xxxxxx
	ALMSG	ALLOCATE	3-3	TYPE CYL CYL
	ALMSG1	ALLOCATE	3-3	....
	ALLEND	FINI	3-3	DEVICE xxx VOLUME xxxxxx ALLOCATION ENCED
	STCYL	FORMALL		ENTER START CYLINDER (xxx) CR "LAEEL":
	ENDCYL	NEXT		ENTER ENC CYLINDER (xxx):
	PROGFOR	REGFORM	3-2	FORMAT STARTED
	RDLAB	LAB	3-1	ENTER DEVICE LABEL:
	ENDFOR	CLEANUP	3-2	FORMAT CONE
	PAGE	CLEANUP	3-2	xxx PAGE RECORDS FLAGGED
	RESULTS	FINI	3-3	ALLOCATION RESULTS
	MAP	PRINTALL		TEMP 000 0C0
	LABELCHK	LABONLY		LAEEL IS NCW xxxxxx

Figure 3-18. The Format/Allocate Program Messages

Chapter 4. DMKDIR—The Directory Program

Introduction

The DMKDIR program builds the VM/370 directory on a volume previously formatted by the Format/Allocate program, using cylinders that were previously allocated for use as directory space.

Under the control of the VM/370 system, the new directory is dynamically swapped and placed in use provided the directory has been created without errors, on a volume in the system-owned list, and provided the user class is A, B, or C.

The new directory can be built so that it does not overlay an existing directory. To do this, allocate enough space for two directories or allocate space for a new directory each time the directory is created.

The directory program can be run standalone or under the control of CMS. The CMS DIRECT command invokes the directory program under CMS.

Method of Operation

This section describes the operation of the VM/370 Directory program. Figure 4-1 shows the relationship of the Method of Operation diagrams.

Diagram 4-1 describes the major functions of the Directory program.

Diagrams 4-2, 4-3, and 4-4 describe the control statement processing and the resulting action.

Diagram 4-5 shows the functions performed before the program terminates.

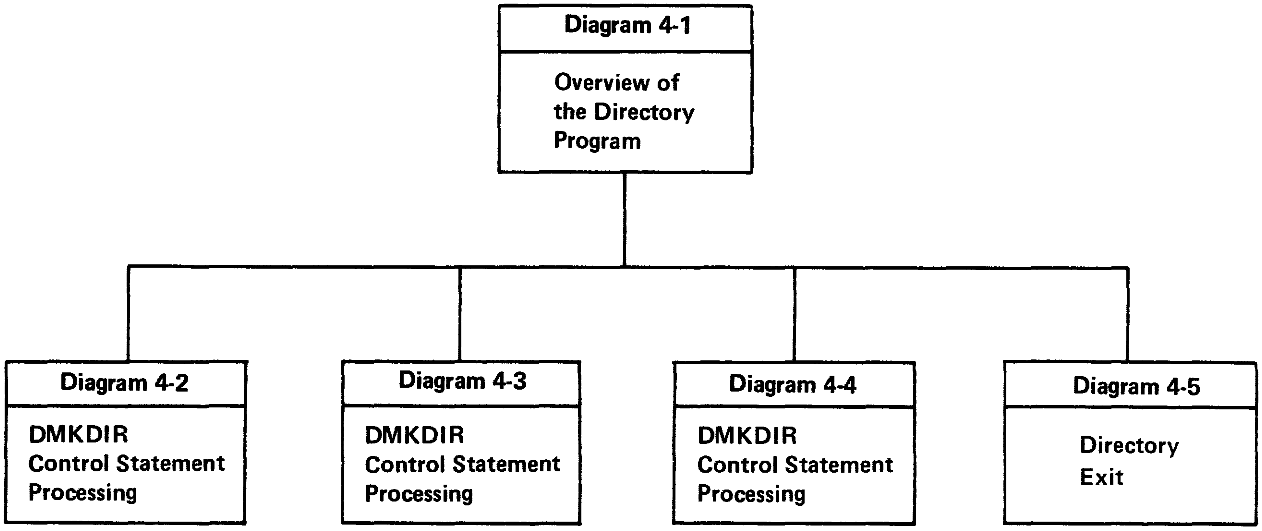
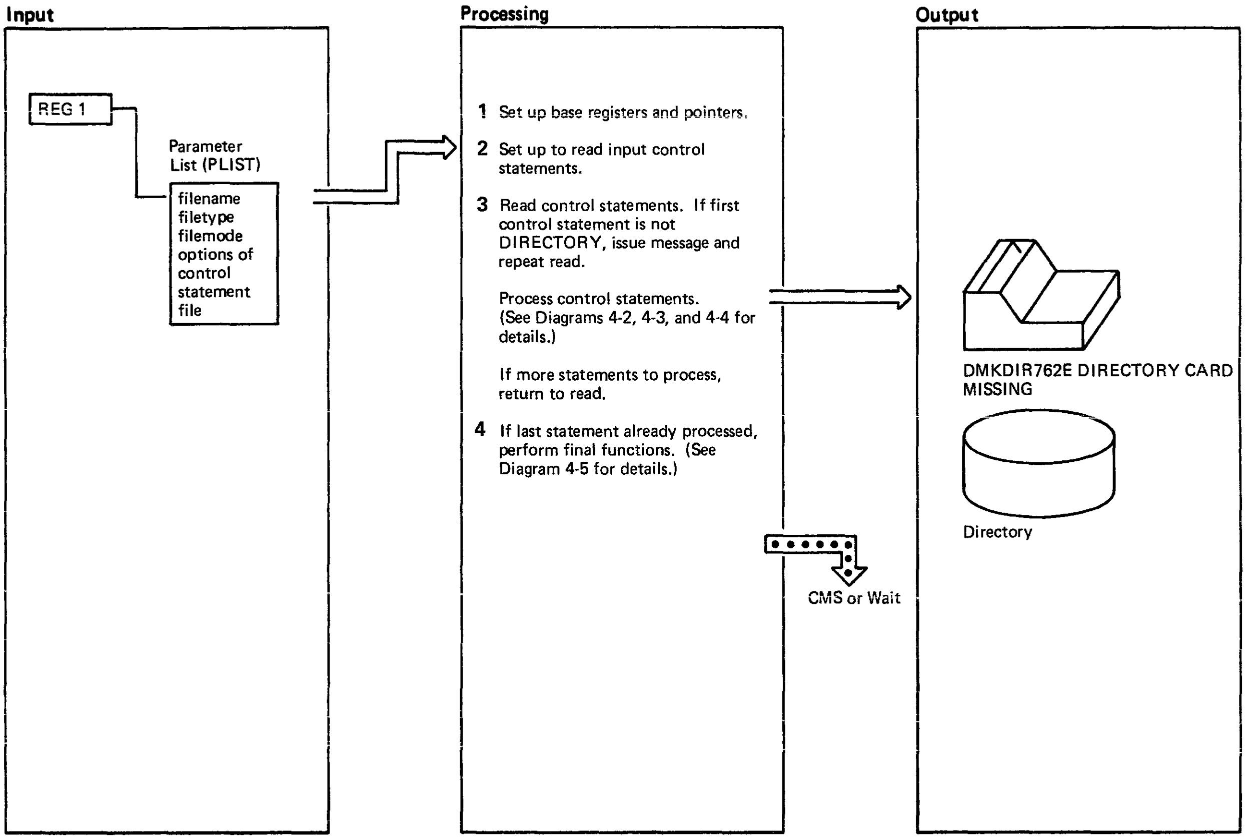


Diagram 4-1
Overview of the Directory Program
Diagram 4-2
DMKDIR Control Statement Processing
Diagram 4-3
DMKDIR Control Statement Processing
Diagram 4-4
DMKDIR Control Statement Processing
Diagram 4-5
Directory Exit

Figure 4-1. Key to the Directory Program Method of Operation Diagrams



1 Set up base registers and pointers.

2 Set up to read input control statements.

3 Read control statements. If first control statement is not DIRECTORY, issue message and repeat read.

Process control statements. (See Diagrams 4-2, 4-3, and 4-4 for details.)

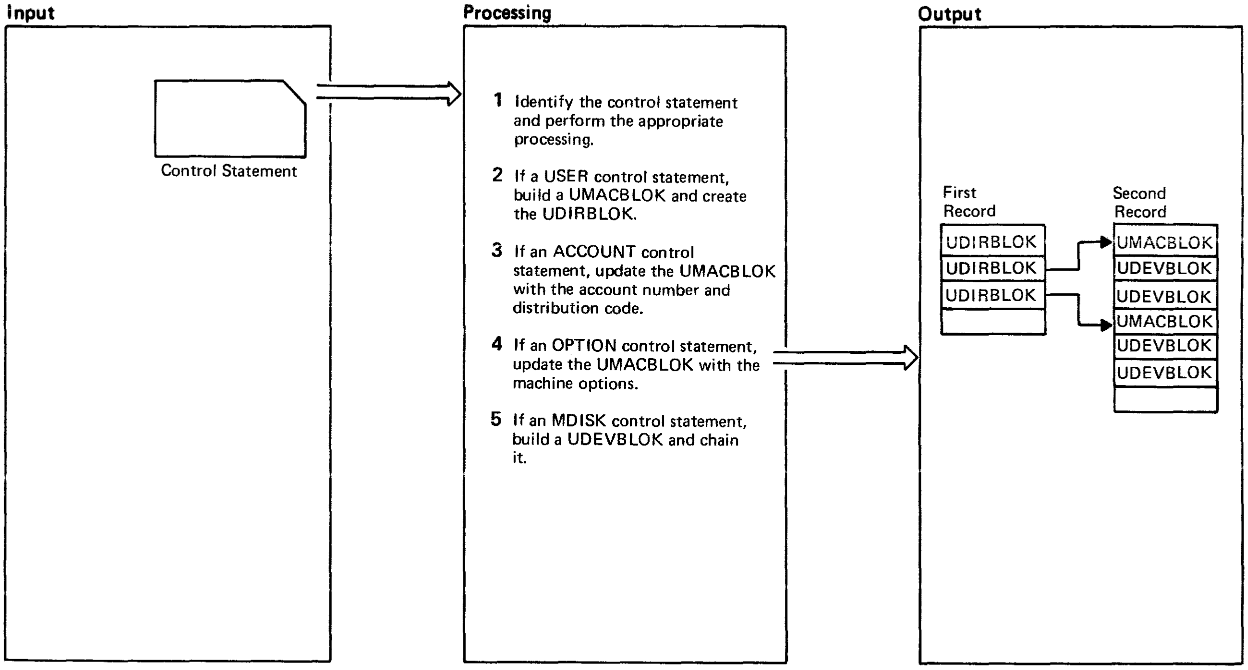
If more statements to process, return to read.

4 If last statement already processed, perform final functions. (See Diagram 4-5 for details.)

Notes	Module	Label	Ref
1 DMKDIR sets up registers 12, 13, and 9 as base registers and sets up pointers to the first UDEVBLOK and the allocation record buffer.	DMKDIR	DMKDIRCT	
2 If running standalone, the header line is printed: VM/370 USER DIRECTORY CREATION PROGRAM RELEASE n ENTER CARD READER DEVICE ADDRESS AND OPTIONS	DMKDIR	MSGRET MSG02A	
The program then reads a response from the console. A read is issued to the card reader indicated (if any). If the operator enters a null line in response to the message, the IPL device is used as the input card reader. If the EDIT option is specified, DIRFLAG is set to X'20'.		DEFAUL13 STOREADD	
If running under CMS, set the P-list containing the filename, filetype and filemode of the file containing the directory control cards. If EDIT is specified, the DIRFLAG is set to X'20'.		CMS 1 EDITTEST	
The STATE macro is issued to see if the control statement file exists. If the file is not found, the messages, DMKDIR763E INVALID FILENAME OR FILE NOT FOUND		STATE	

Notes	Module	Label	Ref
EOJ DIRECTORY NOT UPDATED are displayed and control returns to CMS.		TERM	
3 Control statements are read via SVC 202 when the Directory program is run under the control of CMS. When the Directory program runs standalone, the read function is performed either by the GRAPHID routine (if the console is a display device) or by the STARTIO routine in all other cases. The READ routine scans the control statement and branches to the appropriate processing routine. After processing each control statement and executing the associated routine, control returns to READ to process the next control statement.	DMKDIR	READ GRAPHID STARTIO	
4 When the last statement is read and processed, the READ routine branches and links to the EXIT routine.		EXIT	

Diagram 4-1. Overview of the Directory Program



1 Identify the control statement and perform the appropriate processing.

2 If a USER control statement, build a UMACBLOK and create the UDIRBLOK.

3 If an ACCOUNT control statement, update the UMACBLOK with the account number and distribution code.

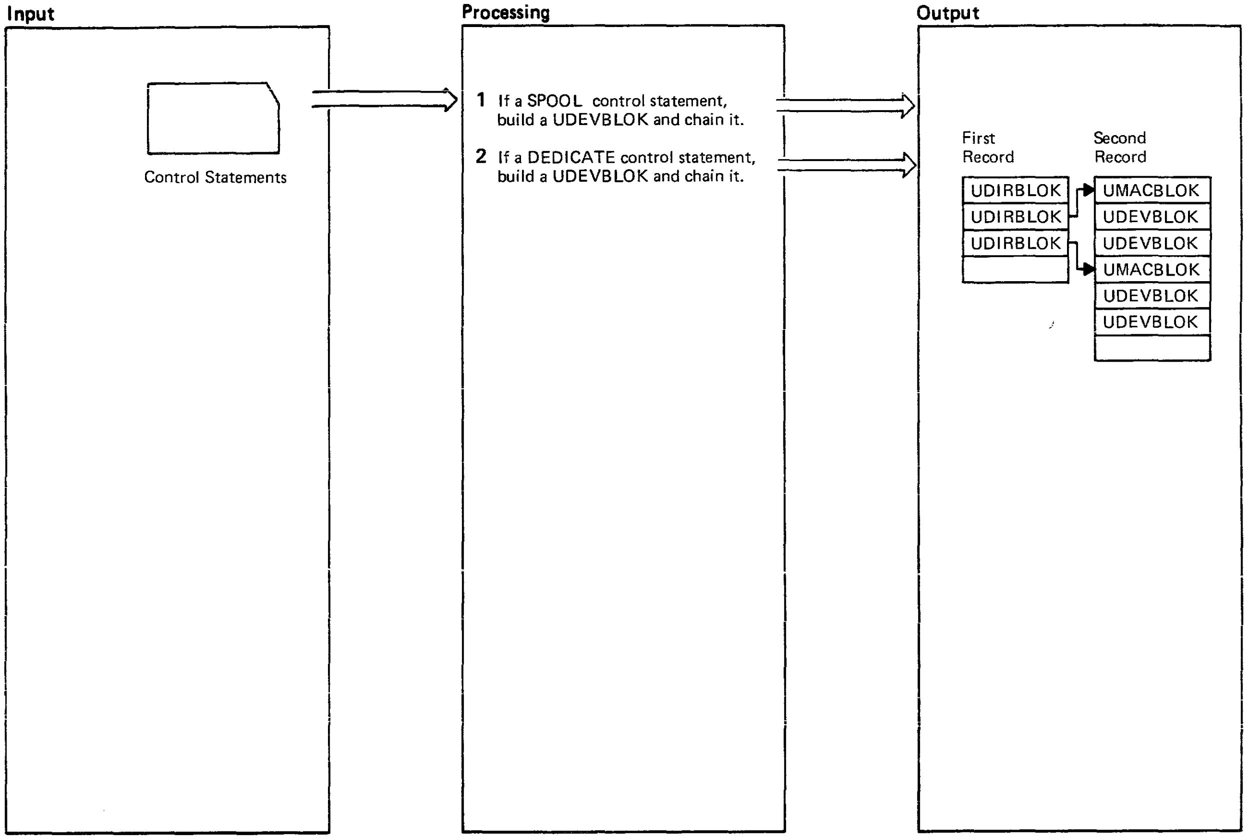
4 If an OPTION control statement, update the UMACBLOK with the machine options.

5 If an MDISK control statement, build a UDEVBLOK and chain it.

Notes	Module	Label	Ref
1 The READ routine branches and links to the SCANNAME routine with register 4 pointing to TABLE1. TABLE1 is searched for a keyword matching the control statement name and control is passed to the routine indicated in the corresponding ADCON.	DMKDIR	READ SCANNAME SCAN1	
2 If the USER control statement follows a USER, ACCOUNT, OPTION, or IPL control statement, DMKDIR752E STATEMENT SEQUENCE ERROR FOLLOWING USER user appears on the console followed by the statement that was out of sequence. Directory processing is terminated after scanning the remaining statements for syntax. The last UDIRBLOK and UMACBLOK are masked off. Update the pointers to the buffers and write out the buffers that are full. The SCANUSER routine locates a UDIRBLOK and initializes it. Then the UMACBLOK is located and initialized.	DMKDIR	SCANUSER ERROR52	
3 If the ACCOUNT control card does not follow a USER, OPTION, or IPL control statement, DMKDIR752E STATEMENT SEQUENCE ERROR FOLLOWING USER user appears on the console followed by the statement that was out of sequence. Directory processing is terminated after scanning the remaining statements for syntax. The SCANACCO routine updates the account number (UMACACCT) and distribution code (UMACDIST) fields of the UMACBLOK.	DMKDIR	SCANACCO ERROR52	

Notes	Module	Label	Ref
4 If the OPTION control statement does not follow a USER or IPL control statement, DMKDIR752E STATEMENT SEQUENCE ERROR FOLLOWING USER user appears on the console followed by the statement that was out of sequence. Directory processing is terminated after scanning the remaining statements for syntax. The SCANOPTI routine sets fields in the UMACBLOK to indicate the machine options.	DMKDIR	SCANOPTI ERROR52	
5 The SCANMDIS routine branches and links to the SCANNAME routine with register 4 pointing to TABLE4. TABLE4 is scanned by device type to get the corresponding device class. The SCANMDIS routine then updates the device type (UDEVTYPE) and class (UDEVTYPC) fields in the UDEVBLOK. The UDEVSTAT field is updated to indicate a T-disk or long block, if either is present, and the number of cylinders is updated. For all disks other than T-disk, the volume serial number, mode, and password field of the UDEVBLOK are initialized. The mode is updated (except for a T-disk). Label Value Comments UDEVR 00 R link-mode UDEVRR 04 RR link-mode UDEVW 08 W link-mode UDEVWR 12 WR link-mode UDEVM 16 M link-mode UDEVMR 20 MR link-mode UDEVMW 24 MW link-mode The SCANMDIS routine then branches to the CHAINDEV routine to chain the UDEVBLOK to the UMACBLOK.	DMKDIR	SCANMDIS SCANNAME CHAINDEV	

Diagram 4-2. DMKDIR Control Statement Processing



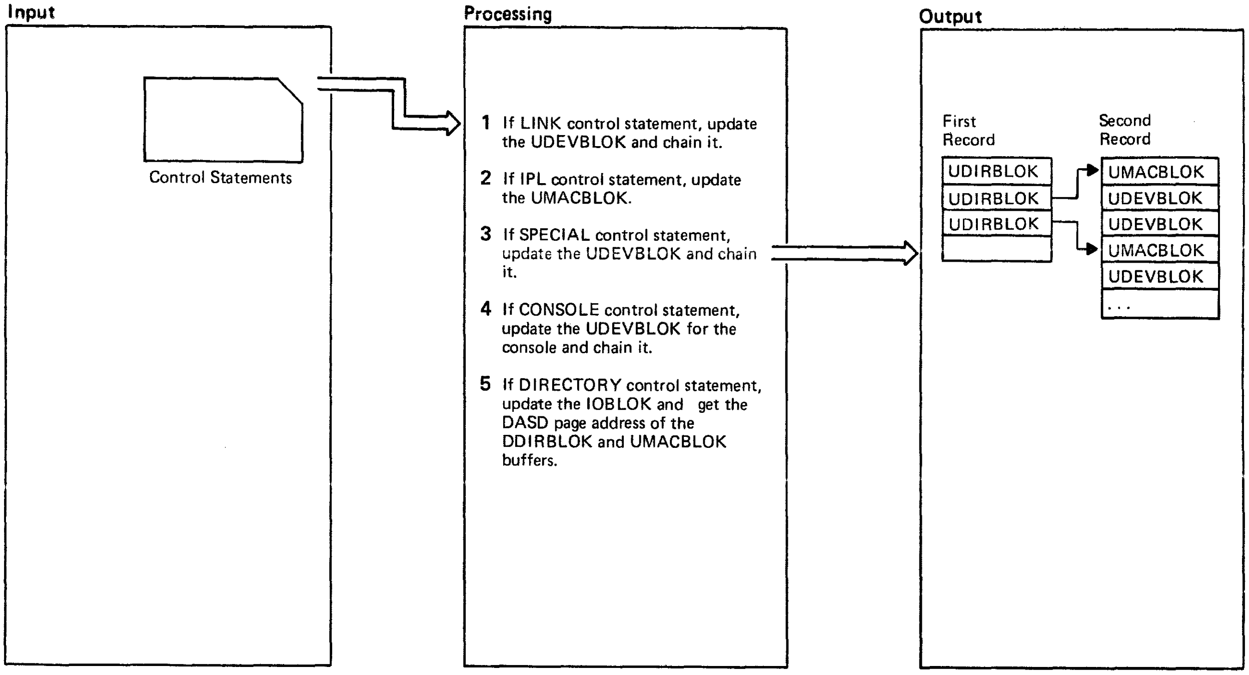
1 If a SPOOL control statement, build a UDEVBLOK and chain it.

2 If a DEDICATE control statement, build a UDEVBLOK and chain it.

Notes	Module	Label	Ref
1 The SCANSPOO routine builds a UDEVBLOK. The UDEVSTAT field is set to X'08' to indicate a spool device. The virtual device address is stored in the UDEVADD field and the spool class is stored in the UDEVCLAS field. The SCANSPOO routine branches and links to the SCANNAME routine with register 4 pointing to TABLE5. For all device types except the 2540, the spool class is picked up directly from TABLE5. For a 2540 device, the device class is determined in the SCAN2540 routine. The default class is A, except for readers (readers default to class*).	DMKDIR	SCANSPOO	
The SCANSPOO routine then branches to the CHAINDEV routine to chain the UDEVBLOK to the UMACBLOK.		CHAINDEV	
2 The SCANDEDI routine builds a UDEVBLOK. The UDEVSTAT field is set to X'80' to indicate a dedicated device. The virtual device address is stored in UDEVADD field. And, either the volume serial number (UDEVVSER) or user link to disk (UDEVLINK) fields are updated.	DMKDIR	SCANDEDI	
The SCANDEDI routine then branches to the CHAINDEV routine to chain the UDEVBLOK to the UMACBLOK.		CHAINDEV	

Notes	Module	Label	Ref

Diagram 4-3. DMKDIR Control Statement Processing



1 If LINK control statement, update the UDEVBLOK and chain it.

2 If IPL control statement, update the UMACBLOK.

3 If SPECIAL control statement, update the UDEVBLOK and chain it.

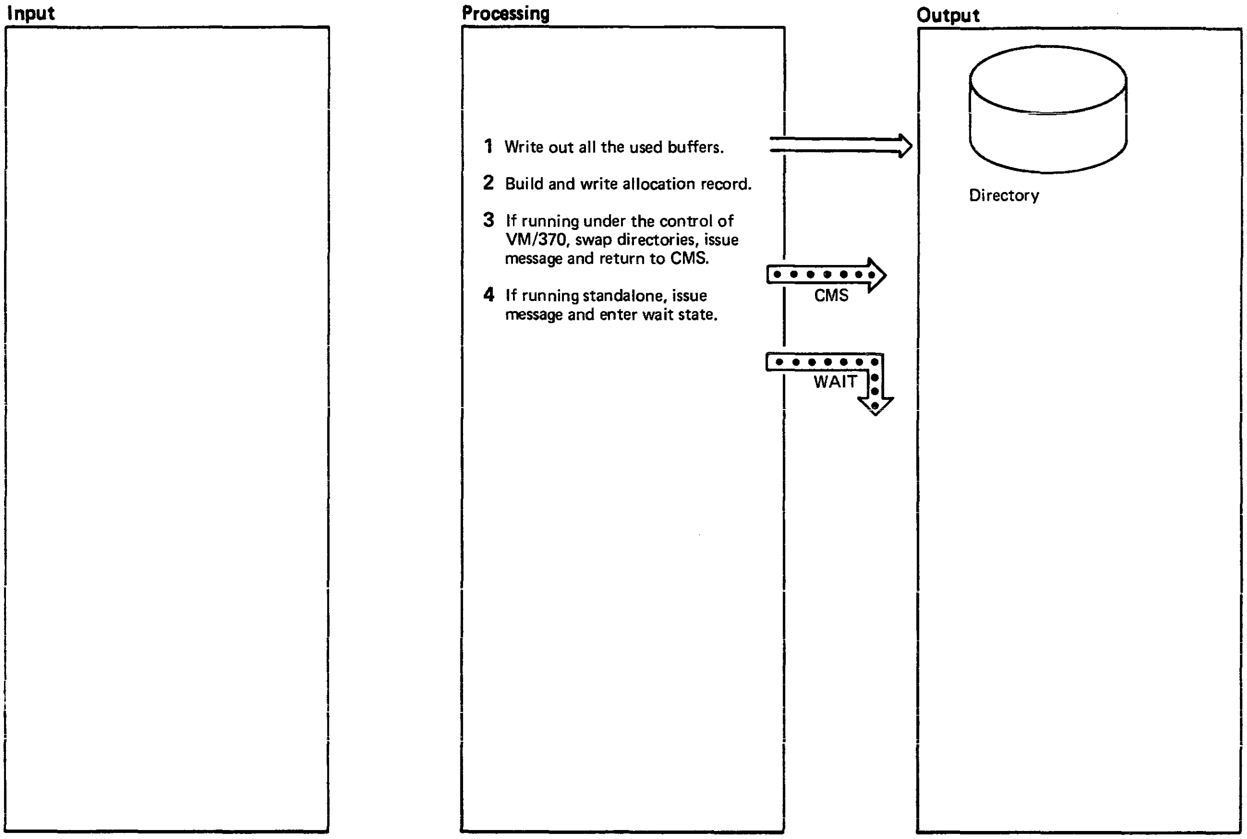
4 If CONSOLE control statement, update the UDEVBLOK for the console and chain it.

5 If DIRECTORY control statement, update the IOBLOK and get the DASD page address of the DDIRBLOK and UMACBLOK buffers.

Notes	Module	Label	Ref
1 The SCANLINK routine builds a UDEVBLOK. The UDEVSTAT field is set to X'10' to indicate the device is to be linked at logon time. The virtual device address (UDEVADD) and link device address (UDEVLINK) are updated. The mode (UDEVMODE) is also updated. Label Value Comments UDEVR 00 R link-mode UDEVRR 04 RR link-mode UDEVW 08 W link-mode UDEVWR 12 WR link-mode UDEVM 16 M link-mode UDEVMR 20 MR link-mode UDEVMW 24 MW link-mode	DMKDIR	SCANLINK	
2 If the IPL control statement does not follow a USER, ACCOUNT, or OPTION control statement, DMKDIR752E STATEMENT SEQUENCE ERROR FOLLOWING USER user appears on the console followed by the statement that was out of sequence. Directory processing is terminated after scanning the remaining statements for syntax. The name of the system to be loaded (via IPL) at logon time is placed in the UMACIPL field of the UMACBLOK.	DMKDIR	SCANIPL ERROR52	
3 The SCANSPEC routine builds a UDEVBLOK for a special device. The virtual device address is stored in UDEVADD, the device type is stored in UDEVTYPE, and the class is stored in UDEVTYPC. The SPANSPEC routine branches and links to the SCANNAME routine with register 4 pointing to TABLE2. The device type and class is picked up directly from TABLE2 for a 3270 and pseudo-timer.	DMKDIR	SCANSPEC SCANNAME SCANCTCA SCAN2701 SCAN2702 SCAN2703	

Notes	Module	Label	Ref
The SCANNAME routine branches (via an ADCON) to the SCANCTCA, SCAN2701, SCAN2702, and SCAN2703 routines to determine the device type and class of channel-to-channel adapter, or 2701, 2702, and 2703 special device.			
4 The SCANCONS routine builds a UDEVBLOK for the console. The virtual device address is stored in UDEVADD, the device type is stored in UDEVTYPE, and the class is stored in UDEVTYPC. The default class is T. The SCANCONS routine branches and links to the SCANNAME routine with register 4 pointing to TABLE3. The device type and class are picked up directly from TABLE3.	DMKDIR	SCANCONS	
5 If the DIRECTORY control statement is not the first control statement, DMKDIR752E STATEMENT SEQUENCE ERROR FOLLOWING USER user appears on the console followed by the statement that was out of sequence. Directory processing is terminated after scanning the remaining statements for syntax. The SCANDIRE routine sets up to update the IOBLOK. The output device address is stored in DASDADD, and the serial number is stored in DASDVSER. The DIRFLAG is set to a hexadecimal value that indicates the device type of the output unit. Then, the SCANDIRE routine gets the pointer to the first page of the directory and machine buffer areas.	DMKDIR ERROR52	SCANDIRE	

Diagram 4-4. DMKDIR Control Statement Processing



1 Write out all the used buffers.

2 Build and write allocation record.

3 If running under the control of VM/370, swap directories, issue message and return to CMS.

4 If running standalone, issue message and enter wait state.

Notes	Module	Label	Ref
1 All of the user directory, user machine, and user device buffers that were used are written. The buffers are written out by loading the DASD address into register 2, loading the buffer address into register 1, and then branching and linking to the WRITE routine.	DMKDIR	EXIT	
2 The allocate table is built. A table setting of X,04' indicates an unallocated cylinder and X'0C' indicates an allocated cylinder. The VOL1 and allocation records are written.	DMKDIR	SCANALLO	
3 First the return PSW is set up and Registers 1 and 2 are set to the volume serial number. The user directories are swapped via a DIAGNOSE call to DMKUDRDS. The DIAGNOSE will program check if the user is not class A, B, or C. The directories are not swapped if the volume is not found in the OWNDLIST or if an I/O error occurs under CP. The message EOJ DIRECTORY UPDATED appears on the console and control returns to CMS. If no errors occur, and if the active system directory was updated, the directories are swapped. The message EOJ DIRECTORY UPDATED AND ON LINE appears on the console and control returns to CMS.	DMKDIR	MOVEPSW LOOP 11	

Notes	Module	Label	Ref
4 If not running under VM/370, the message EOJ DIRECTORY UPDATED appears on the console and the wait state is entered by loading the SVCNEW PSW.	DMKDIR	BARE	

Diagram 4-5. Directory Exit

Program Organization

This section includes a program description of the DMKDIR module.

DMKDIR

Creates the VM/370 directory on a system owned volume.

Entry Points

DMKDIRCT is the entry point when the directory program is executed standalone and DMKDIRED is the entry point when the directory program is executed under the control of CMS.

Routines Called

None

Attributes

Not serially reusable.

Registers at Exit

If executed under the control of CMS, register 15 contains a return code at exit.

Return Code	Meaning
1	Invalid filename or file not found.
2	Error loading the directory.
3	Invalid option from CMS.
4	Directory not swapped, user class not A, B, or C.
5	Directory not swapped, system (cld) directory locked.
6	Directory not swapped; the directory the system is using is not the directory just updated.
1xx	Error in CMS RDBUF routine.
2xx	Error in CMS TYPLIN routine.

where xx is the CMS routine return code.

Register Usage

R0:	Work register.
R1:	Pointer to input field. Pointer to IOB. Pointer to output buffer. Work register.
R2:	Input count from SCANCARD. DASD address. Work register.
R3:	Work register.
R4:	Work register.
R5:	Branch and link return address. Pointer to the next UDEVBLOK. Work register.
R6:	RDIRBUF, pointer to the UDIRBLOK buffer.
R7:	RMACBUF, pointer to the UMACBLOK buffer.
R8:	RDEVBUF, pointer to the UMDEVBLOK buffer.
R9:	Base register 3.
R10:	RMAC, pointer to UMACBLOK.
R11:	RDEV, pointer to UDEVBLOK.
R12:	Base register 1.
R13:	Base register 2.
R14:	Return address.
R15:	RDIR, pointer to UDIRBLOK.

External References

DMKURDS is called via a DIAGNOSE instruction to write the new VM/370 directory on DASD.

Directory

Figure 4-2 is an alphabetic list of the major labels cf the Directory program. The associated method of operation diagram is referenced and a brief description of the function performed at the pcint in the program corresponding tc each label is included.

Label	Diagram	Description
BARE	4-5	Directory program exit when not running under the control of VM/370.
BILDUDIR		Builds UDIRBLOK.
BILDUMAC		Builds UMACBLOK.
BINCONV		Converts decimal numbers to binary.
CHAINDEV	4-2,4-3	Chains UDEVBLOK to UMACBLCK.
CLEARUDR		Clears UDIRBLOK.
CMS1	4-1	Sets up the parameter list identifying the file ccntaining the control statements when running under CMS.
CMS3		Reads CMS control cards via SVC 202.
COMPARE		Compares keywords and sets condition codes.
DECCONV		Converts decimal numbers to hexadecimal.
DEFAUL13	4-1	Defaults to the IPL device for control statement input device when running standalone.
DMKDIRCT	4-1	Sets up base registers and initializes pointers.
EDITTEST	4-1	Sets DIRFLAG to X'20' to indicate edit, if EDIT is specified when the Directory program is run under VM/370.
EOF		Simulates a USER card.
ERROR51		Error processing for invalid operand.
ERROR52	4-2 4-4	Issues message when a control statement is out cf sequence.
ERROR58		Issues message DMKDIR758E.
ERROR62		Issues message DMKDIR762E.
EXIT	4-1 4-5	End-of-job processing for Directory Program.
GETCARD		Reads control statement in input buffer.
GETCYLNO		Fills in cylinder relocation for minidisks.
GETPAGE		Assigns a DASD page address.
GRAPHID	4-1	Reads the input control statements from a display terminal when the directory program is not running under CMS.
HEXCONV		Converts hexadecimal numbers to binary.
LONG		Turns on long block indicator for minidisks.
LOOP11	4-5	Calls DMKUDRDS via the DIAGNOSE instruction to swap directories when running under VM/370.
MOVECPT		Sets up current control statement pointer.
MOVEDISP		Updates UMACBLOK.
MOVEPSW	4-5	Sets up return PSW before issuing DIAGNOSE to call DMKUDRDS.
MSGRET	4-1	When running standalone, a header line is printed.
MSG02A	4-1	Requests input device when running standalone.
MSGWRITE		Writes messages to the terminal.
NOTUSED		Updates UMACBLOK pointer.
POINTDEV		Updates UDEVBLOK pointer.

Figure 4-2. The Directory Program Label Directory (Part 1 of 2)

Label	Diagram	Description
READ	4-1 4-2	Reads control statements and branches to appropriate processing routine.
READCARD		Reads control statements.
REREAD		Sets up pointer to control statement read buffer.
RET1		Scans control statements.
SCANACCO	4-2	ACCOUNT statement processing routine.
SCANALLO	4-5	Builds allocation record.
SCANCARD		Scans the control statement for the next operand.
SCANCONS	4-4	CONSOLE statement processing routine.
SCANCTCA	4-4	Updates the UDEVBLOK and chains the control unit to the UDEVBLOK for channel-to-channel adapters.
SCANDEDI	4-3	DEDICATE statement processing routine.
SCANDIRE	4-4	DIRECTORY statement processing rcutine.
SCANIPL	4-4	IPL statement processing routine.
SCANLINK	4-4	LINK statement processing routine.
SCANMDIS	4-2	MDISK statement processing routine.
SCANNAME	4-2 4-4	Scans the name table until a match is found. Register 4 points to the name table. If the name field is a constant, it is put in the UDEVBLOK. If the name field is an address, control is passed to that address.
SCANOPTI	4-2	OPTION statement processing routine.
SCANSPEC	4-4	SPECIAL statement processing routine.
SCANSPOO	4-3	SPOOL statement processing routine.
SCANUSER	4-2	USER statement processing routine.
SCAN1	4-2	Points register 4 to TABLE1, then branches and links to SCANNAME routine to determine the appropriate control statement processing routine.
SCAN2311		Updates the UDEVBLOK for 2311 disks.
SCAN2540		Updates the UDEVBLOK for 2540 devices.
SCAN2701	4-4	Updates the UDEVBLOK for 2701 devices.
SCAN2702	4-4	Updates the UDEVBLOK for 2702 devices.
SCAN2703	4-4	Updates the UDEVBLOK for 2703 devices.
STARTIO	4-1	Reads the input control statements if the directory program is not running under CMS.
STATE	4-1	Checks that control statement file exists.
STOREADD	4-1	Sets the DIRFLAG to X'20' to indicate edit, if EDIT is specified when the Directory program is run standalone.
TERM	4-1	At end of processing, returns control to CMS if running under VM/370.
TESTBUFF		Tests to see if UDEVBLOK was used.
TESTUDEV		Gets DASD address of UMACBLOK.
UPDATE		Points to next UDEVBLOK.
UPDATECT		Updates device count in UMACBLOK.
WRITE		Writes the directory on DASD.

Figure 4-2. The Directory Program Label Directory (Part 2 of 2)

Data Areas

The directory exists on disk as 4K (page size) records. The VOL1 label (cylinder 0 track 0 record 3), on the volume containing the directory, points to the directory. The directory starts with the first available record.

The first UDIRBLOK is a dummy UDIRBLOK. Its UDIRDISP field points to the last UDIRBLOK in that record. The UDIRDASD field points to the next UDIR record, or, if it is the last record, it contains zeros. The second UDIRBLOK in the first record points to the UMACBLOK for that user, located in the second record. In turn, the UMACBLOK points to the first UDEVBLOK for that user. It is the second block in the second record. The last UDEVBLOK for this user has a pointer of all zeros.

The directory entry for the second user consists of a UDIRBLOK in the first record and associated UMACBLOK, and UDEVBLOKs in the second record. When a record becomes full, the chain continues into the next available record.

When the directory is created, all UDIRBLOKs are grouped 169 blocks per record. The UMACBLOK and UDEVBLOKs are sequentially chained into a separate record. If the record becomes full before the end of the chain, the chain overflows into the next available record.

The formula to find the number of records is:

$$\frac{NU}{169} + \frac{((NU+NM)X2)+ND}{170} = NR$$

where:

NU is the number of user records.

NM is the number of MDISK cards describing a virtual disk (not T-Disk).

ND is the total number of MDISK (describing T-Disk space), SPOOL, LINK, SPECIAL, CONSOLE, and DEDICATE cards.

NR is the total number of records used.

To find the number of cylinders, divide the total number of records by 32 for 2314/2319 devices, by 57 for 3330 series devices, or by 24 for 3340 and 2305 series devices. To ensure that a new directory will not overlap an existing directory, allow space for two directories or allocate a new directory each time the directory is created.

The following data areas are used by the directory program:

- The UDEVBLOK (user device block), built in the UDEVBLOK or UMACBLOK buffer.
- The UDIRBLOK (user directory block), built in the DIRBLOK buffer.
- The UMACBLOK (user machine block), built in the UMACBLOK buffer.

These data areas, as well as a figure showing the user directory format and the relationship of the above blocks, are described in the *VM/370 Data Areas and Control Blocks Logic*.

Diagnostic Aids

Figure 4-3 lists the messages issued by the Directory program. The label of the message and the associated method cf operation diagram are included in the list.

Message Code	Label	Diagram	Message Text
DMKDIR751E	ERROR51A		INVALID OPERAND - xxxxxxxxxxxx.
DMKDIR752E	ERROR52	4-2,4-4	STATEMENT SEQUENCE ERROR FOLLOWING USER user
DMKDIR753E	ERROR53		OPERAND MISSING
DMKDIR754E	ERROR54A STARTIO READ WRITE		DEV ccu NOT OPERATIONAL
DMKDIR755E	ERROR55A WRITE		IO ERROR ccu CSW xxxxxxxxxxxxxxxx SENSE xxxxxxxxxxxx
DMKDIR756E	ERROR56A		PROGRAM CHECK PSW = xxxxxxxxxxxxxxxx
DMKDIR757E	ERROR57		MACHINE CHECK RUN SEREP AND SAVE OUTPUT FOR CE
DMKDIR758E	ERROR58 CHAINDEV		DUPLICATE UNIT DEFINITION
DMKDIR760E	ERROR60 GETPAGE		NOT ENOUGH SPACE ALLOCATED FOR DIRECTORY
DMKDIR761E	ERROR61A SCANDIRE		VOLID READ IS volid1 NOT volid2
DMKDIR762E	ERROR62 READ		DIRECTORY STATEMENT MISSING
DMKDIR763E	ERROR63 STATE	4-1	INVALID FILENAME OR FILE NOT FOUND
DMKDIR764E	ERROR64		ERROR IN xxxxxxxx
	MSG04	4-1	EOJ DIRECTORY NOT UPDATED
	MSG01	4-5	EOJ DIRECTORY UPDATED
	MSG03	4-5	EOJ DIRECTORY UPDATED AND ON LINE
	MSG02	4-1	VM/370 USER DIRECTORY CREATION PROGRAM RELEASE n
	MSG02A	4-1	ENTER CARD READER DEVICE ADDRESS AND OPTIONS

Figure 4-3. The Directory Program Messages

Chapter 5. The DASD Dump Restore Program

Introduction

The DASD Dump Restore program executes under the control of CMS via the DDR command. It performs five functions for direct access storage devices. The five functions are:

- Dump
- Restore
- Copy
- Print
- Type

DUMP

The dump function saves data from a direct access volume on magnetic tape. The data is saved cylinder by cylinder. The format of the tape is:

- Record 1, volume header record -- data describing the volume.
- Record 2, track header record -- a list of count fields to restore the track and the number of data records written on tape. After the last count field the record contains key and data records to fill the 4K buffer.
- Record 3, track data records -- key and data records packed into 4K blocks with the last block truncated.
- Record 4, either the end-of-volume or end-of-job trailer label. The end-of-volume label contains the same information as the next volume header record except that the ID field contains EOV. The end-of-job trailer label contains the same information as record 1 except that the cylinder number field contains the disk address of the last record on tape and the ID field contains EOJ.

RESTORE

The restore function transfers data from a tape created by the DDR dump function to a DASD device. The data may be restored only to a device of the same type as the device from which it was dumped.

COPY

The copy function copies data from one device to another device of the same type. For disk-to-disk operations, data may be reordered on a cylinder basis. If copying from tape-to-tape, the input tape must have been created by the DDR dump function.

PRINT

The print function prints both hexadecimal and EBCDIC representations of selected records of a DASD or Tape Volume on a printer.

TYPE

The type function displays at the terminal both hexadecimal and EBCDIC representations of selected records of a DASD or tape volume.

Method of Operation

The method of operation diagrams describe the major functions of the DDR (DASD Dump Restore) program. The relationship of the method of operation diagrams is described in Figure 5-1.

Diagram 5-1 describes the major functions of the DDR program.

Diagram 5-2 shows the control statement processing for the DDR program.

Diagram 5-3 describes the Dump function.

Diagram 5-4 describes the Restore function.

Diagram 5-5 describes the Copy function.

Diagram 5-6 describes the Print function.

Diagram 5-7 describes the Type function.

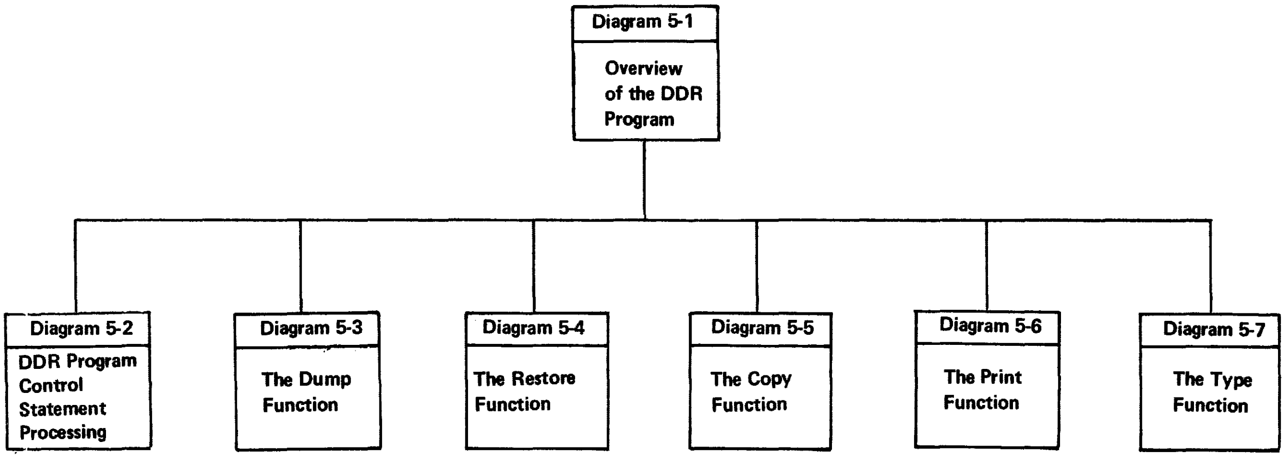
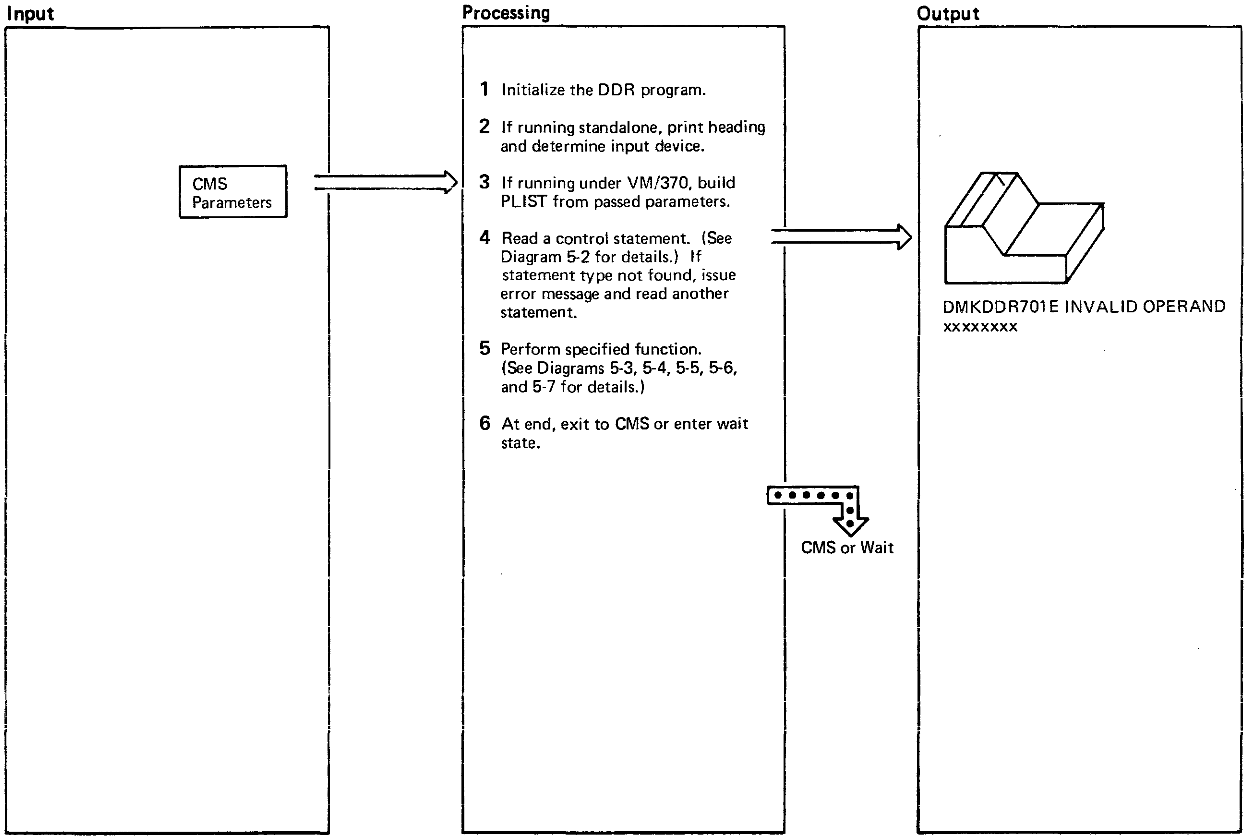


Figure 5-1. Key to the DASD Dump Restore Program Method of Operation Diagrams



1 Initialize the DDR program.

2 If running standalone, print heading and determine input device.

3 If running under VM/370, build PLIST from passed parameters.

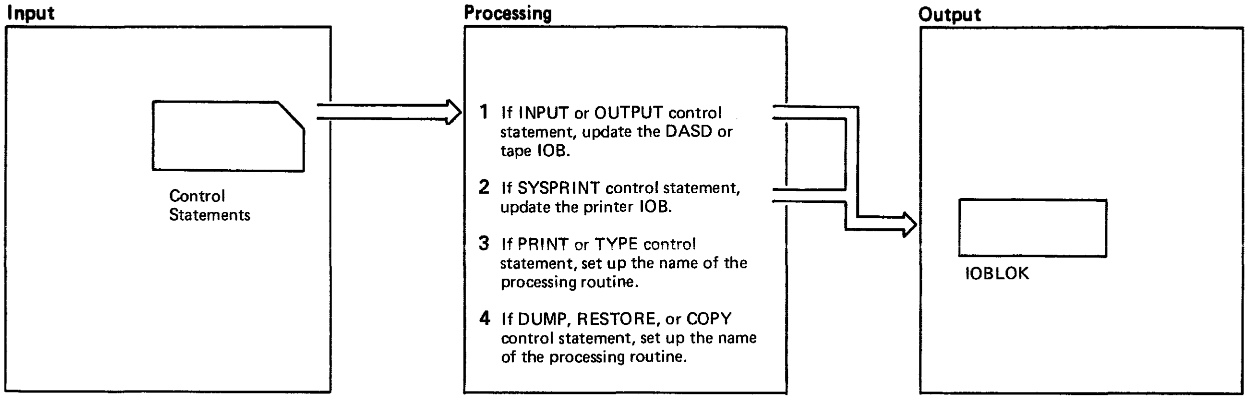
4 Read a control statement. (See Diagram 5-2 for details.) If statement type not found, issue error message and read another statement.

5 Perform specified function. (See Diagrams 5-3, 5-4, 5-5, 5-6, and 5-7 for details.)

6 At end, exit to CMS or enter wait state.

Notes	Module	Label	Ref
1 The DDR program is initialized and the base registers (9, 10, 11, 12, and 13) are set up.	DMKDDR	DMKDDREP	
2 The heading VM/370 DASD DUMP/RESTORE PROGRAM RELEASE n is displayed.	DMKDDR	NEWADD	
If no input device is specified, the IPL device is used as the input device.			
3 DMKDDR builds a PLIST if parameters are passed from CMS to the DDR program.	DMKDDR	CMS1	
4 DMKDDR reads the control statement. The routine needed to initialize the DDR function is found by branching and linking to the SCANNAME routine and searching the name table.	DMKDDR	GTCARD	
5 The designated function is performed. At its end, control returns to the GTCARD routine to read the next control statement and perform the next function.	DMKDDR		
6 When the last control statement is read and processed the GTCARD routine branches to the EXIT routine.	DMKDDR	EXIT	
The end of job statement (MSG001) is displayed. If running under VM/370, the SYSPRINT device is closed and control returns to the CMS command environment.		CMS8	
If running standalone, the wait state is entered.		TESTCMS	

Diagram 5-1. Overview of the DDR Program



1 If INPUT or OUTPUT control statement, update the DASD or tape IOB.

2 If SYSPRINT control statement, update the printer IOB.

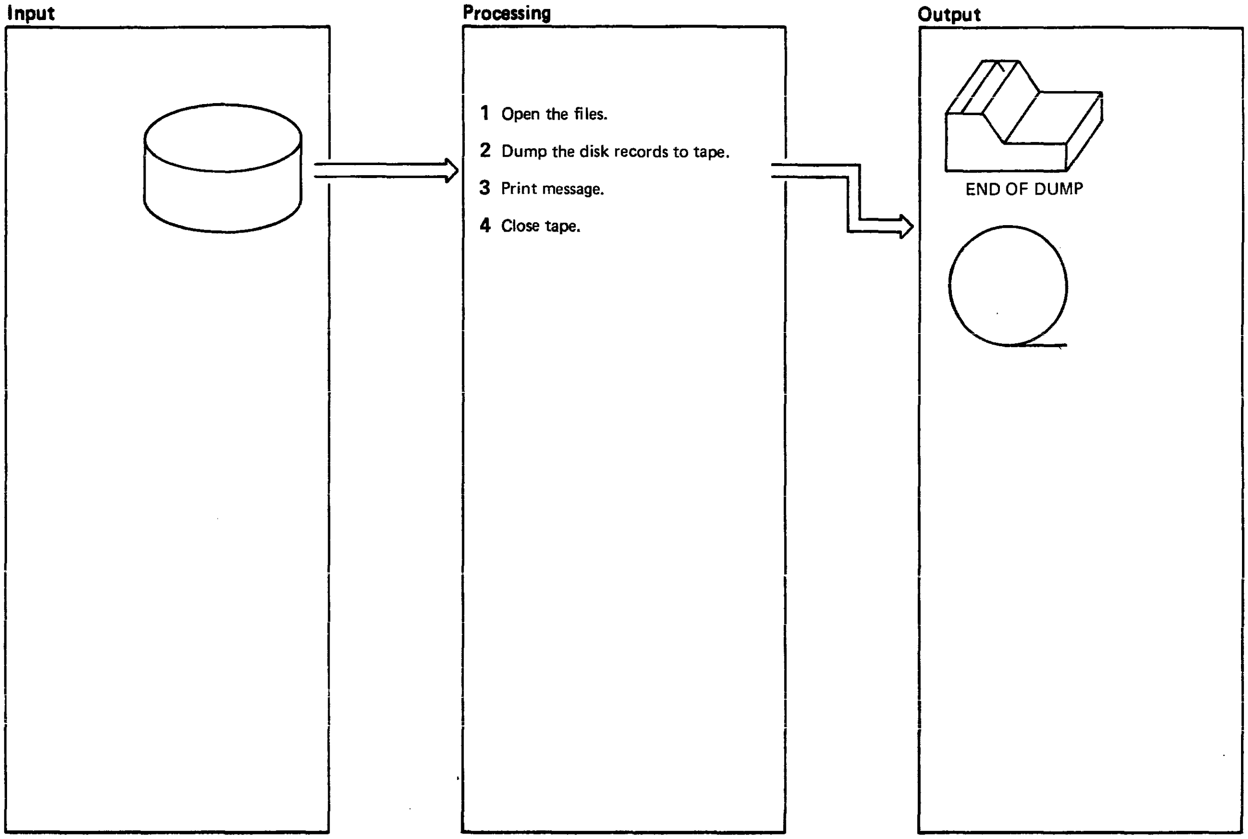
3 If PRINT or TYPE control statement, set up the name of the processing routine.

4 If DUMP, RESTORE, or COPY control statement, set up the name of the processing routine.

Notes	Module	Label	Ref
1 The address of the IOB is loaded into register 15. DMKDDR gets the unit address. The unit address (IOBUADD) and alternate tape address (IOBATAPE) fields of the IOB are filled in. DMKDDR reads the device type from the control statement and then branches and links to the SCAN routine. The SCAN routine searches a table of valid devices and picks up the device class and type. The class (IOBCLASS) and type (IOBTYPE) fields are updated. The codes for the various device classes are contained in the DEVTYPES COPY file. If a DASD serial number is specified, the volume serial number (IOBVSER) field is updated. If tape options are specified, the IOB is updated. Field — Options IOBSKIP — number of times file tobe forward spaced. IOBMODE — tape mode. X'C3' indicates 9-track 1600 BPI X'CB' indicates 9-track 800 BPI IOBDISP — disposition of tape. X'07' indicates rewind X'0F' indicates rewind and unload X'03' indicates tape is not to be repositioned Either of the following error messages may be displayed while processing INPUT o OUTPUT control statements: DMKDDR701E INVALID OPERAND – xxxxxxxx DMKDDR703E OPERAND MISSING If either of these errors occurs, the control statement is ignored and control returns to the GTCARD routine to read the next control statement.	DMKDDR	SCANINPU SCANOUTP	
2 The address of the printer IOB is loaded into register 15. The printer unit address is placed in the IOBUADD field of the IOB.	DMKDDR	SCANSYSP	

Notes	Module	Label	Ref
If an error occurs, either message DMKDDR701E INVALID OPERAND – xxxxxxxx DMKDDR703E OPERAND MISSING is displayed. The statement in error is ignored, and control returns to the GTCARD routine to read the next control statement.			
3 The translate table is set up. If TYPE is specified, the LOWERCAS table is used. If PRINT is specified, the UPPERCAS table is used. The routine name is set up: PRINT or TYPE. The start address (default is track 0 record 0) and the stop address (default is last track and last record) are set up. If TYPE is specified, the console skips one line. If PRINT is specified, the printer skips to channel 1. If there is an error in the control statement, either error message DMKDDR701E INVALID OPERAND– xxxxxxxx DMKDDR703E OPERAND MISSING is displayed. The control statement is ignored, and the next control card is read by the GTCARD routine.	DMKDDR	SCANPRIN SCANTYPE	
4 If DUMP control statement, set the processing routine name to DUMP.	DMKDDR	SCANDUMP	
If RESTORE control statement, set the processing routine name to RESTORE.		SCANREST	
If COPY control statement, set the processing routine name to COPY. For the dump function, the input must be a DASD and output a tape. For the restore function, the input must be a tape, the output a DASD. For a copy function, the input and output devices must be the same class and type. If the input device contains more cylinders than the output device the following message is issued: DMKDDR725R ORIGINAL INPUT DEVICE WAS(IS) LARGER THAN OUTPUT DEVICE The operator must determine if the copy function is to continue.		SCANCOPY	

Diagram 5-2. DDR Program Control Statement Processing



1 Open the files.

2 Dump the disk records to tape.

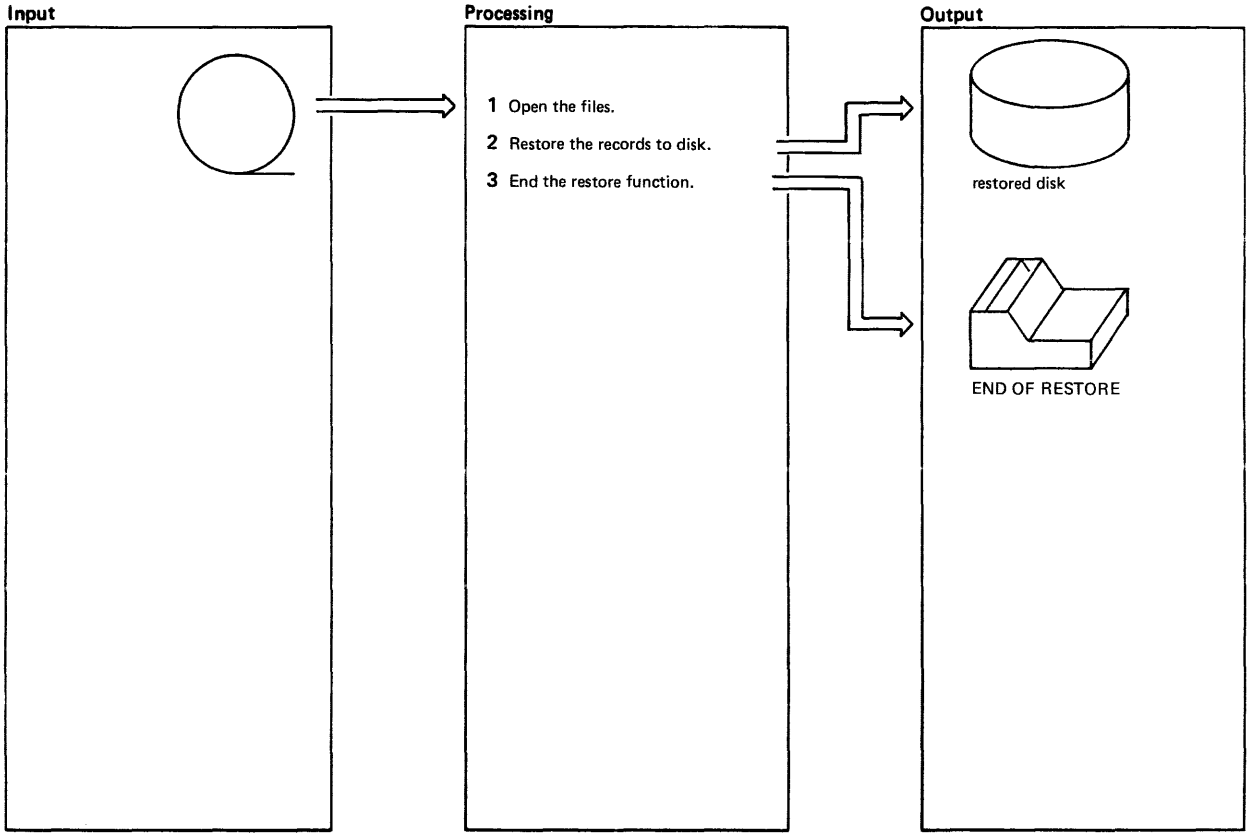
3 Print message.

4 Close tape.

Notes	Module	Label	Ref
1 The input disk is opened by branching and linking to the OPENDASD routine. The extent table is updated to define the cylinders to be dumped. Each statement updates the extent table until a null line, an INPUT statement, or OUTPUT statement is read.	DMKDDR	OPENIN GETEXT	
The output tape is opened, the proper number (if any) of records is skipped and the volume header record (VHR) is written.		OPENOUT	
2 Prints the headings indicating the function being performed and the date and time of the dump.	DMKDDR	PRINTH MSG004	
The read, write, and update cycle continues until the indicated disk extents are dumped to tape. Starting at the first disk extent (CYLSTART), the disk records are read. The record is written on tape and the pointers are updated to the next disk record. The dump cycle continues until the last disk extent CYLSTOP is dumped to tape.		BUILDTHR TESTOUT UPDTADD	
3 The message END OF DUMP indicates that the dump function has successfully terminated.	DMKDDR	CLOSEJOB	

Notes	Module	Label	Ref
4 The trailer record is written on the output tape. If the tape disposition was specified on the DUMP control statement, the tape is so positioned now. Control returns to the control statement read routine (GTCARD) to read and process the next control statement.	DMKDDR	EOJ	

Diagram 5-3. The Dump Function



1 Open the files.

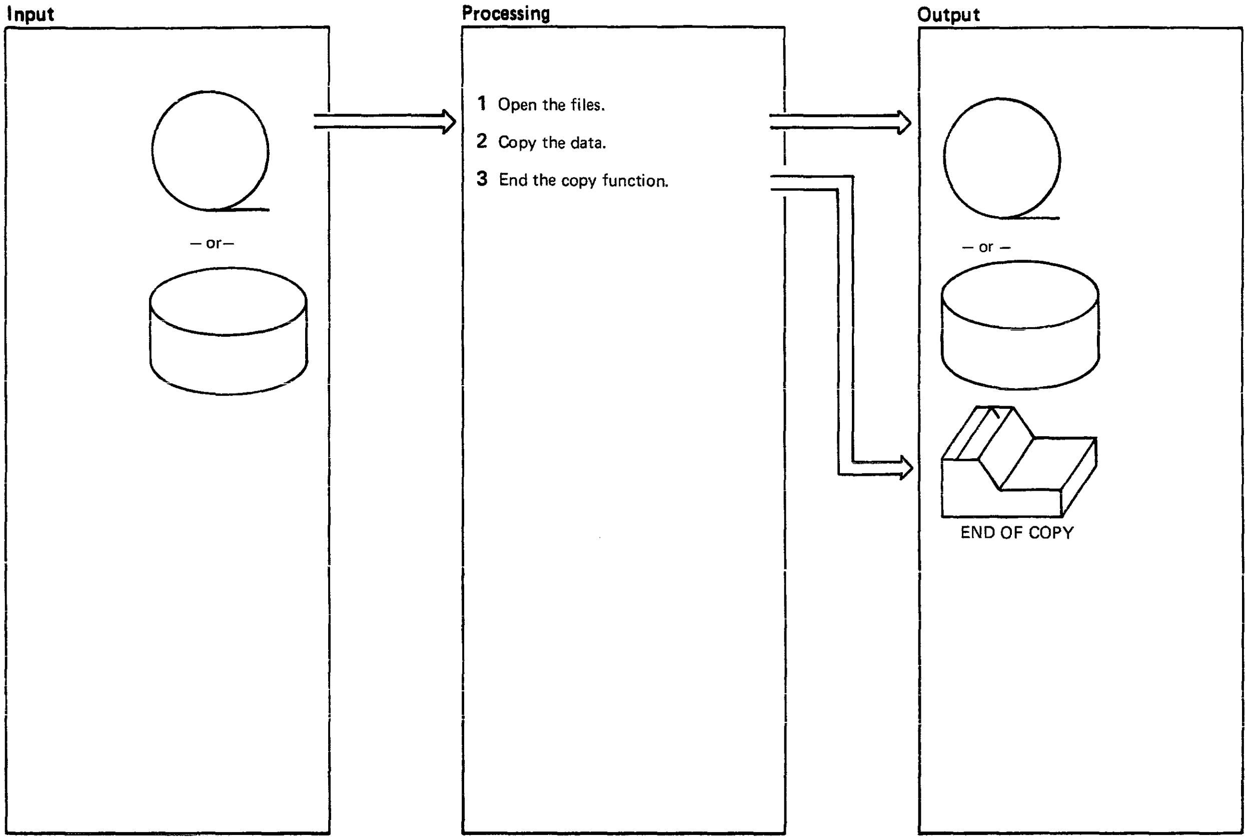
2 Restore the records to disk.

3 End the restore function.

Notes	Module	Label	Ref
1 The input tape is opened and positioned if the RESTORE control statement specified that records were to be skipped.	DMKDDR	OPENIN	
A check is made to ensure that the output disk has the correct volume serial number. If the volume serial number is incorrect, the message DMKDDR717R DATA DUMP FROM xxxxxx TO BE RESTORED TO xxxxxx is displayed. The operator must decide if the restore function is to continue.		SETDASD	
The extent table is updated to indicate the cylinders to be restored to disk.		GETEXT	
The output disk is opened by branching and linking to the OPENDASD routine.		OPENOUT	
2 The headings are printed, indicating that the restore function is starting.	DMKDDR	PRINTD MSG004	
The number of cylinders on the original DASD input device is compared with the number of cylinders on the DASD output device. If the input device was larger, the following message is issued: DMKDDR725R ORIGINAL INPUT DEVICE WAS(IS) LARGER THAN OUTPUT DEVICE			

Notes	Module	Label	Ref
The operator must determine if the restore function is to continue.			
The read and write loop continues until all the specified cylinders are restored to disk. The tape records are read from the tape that has been positioned. The data is written on on the indicated disk cylinders and the pointers to the disk are updated for the next record. The restore function is complete when the last cylinder (CYLSTOP) is restored.		GETTHR DASDWRIT UPDTADD	
3 The message END OF RESTORE is displayed and control returns to the GTCARD routine to read the next control statement.	DMKDDR	CLOSEJOB	

Diagram 5-4. The Restore Function



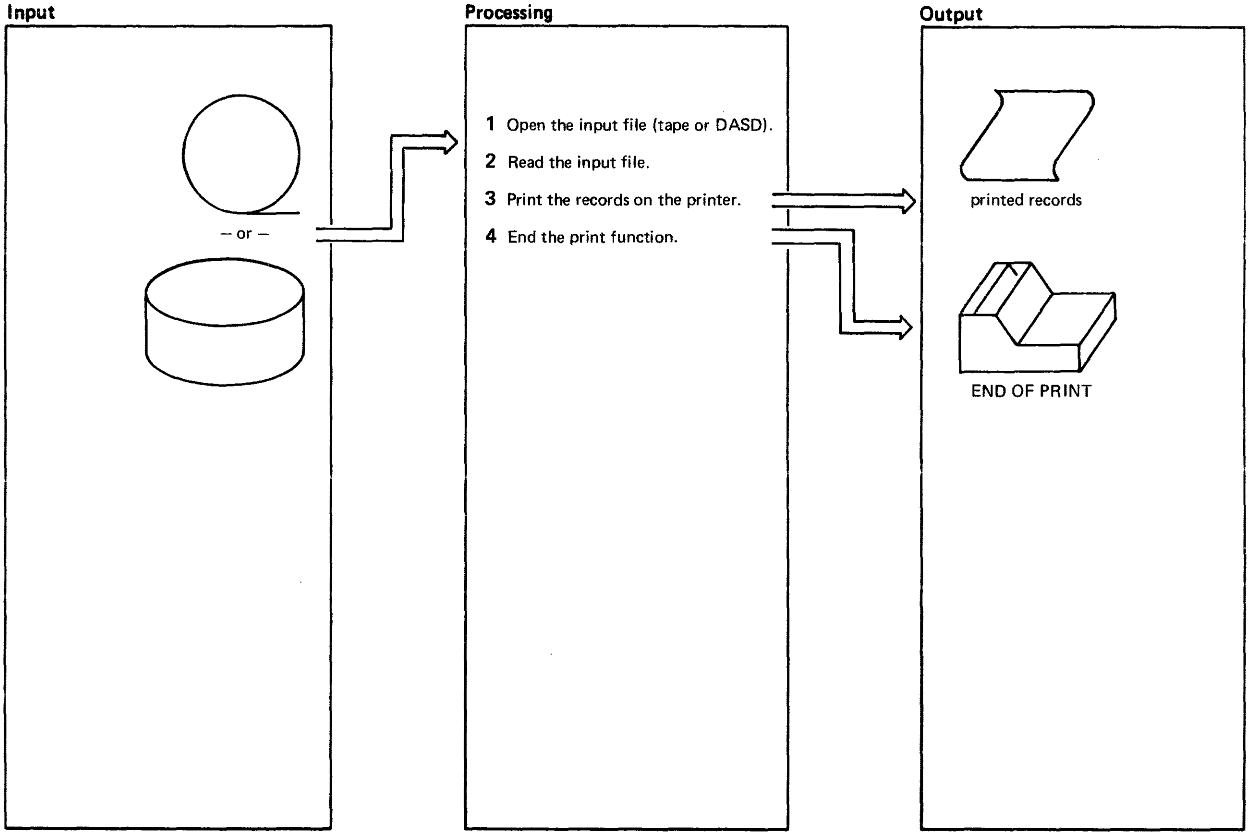
1 Open the files.

2 Copy the data.

3 End the copy function.

Notes	Module	Label	Ref
1 The input file and output file are opened. The input and output devices must be the same device type. The extent table is updated to reflect the amount of data to be copied from one device to another.	DMKDDR	OPENIN GETEXT OPENOUT	
2 The heading is written and the message indicating the start of the copy function is typed. The input file is read and the output file is written. If copying from disk to disk the pointers to the disk records are updated to the next record. The read-write cycle continues until the specified data is copied. When copying data from tape to tape, the GETTHR routine performs the record read and the TESTOUT routine performs the record write. When copying data from disk to disk, the BUILDTHR routine performs the record read and the DASDWRIT routine performs the record write.	DMKDDR	PRINTH MSG004 UPDTADD	
3 The message END OF COPY indicates the successful completion of the copy function. When copying data from tape to tape, the output tape is positioned as indicated on the COPY control card. When the disk to disk copy is complete, the disk file is closed. Control returns to the GTCARD routine to read the next control statement.	DMKDDR	CLOSEJOB	

Diagram 5-5. The Copy Function



1 Open the input file (tape or DASD).

2 Read the input file.

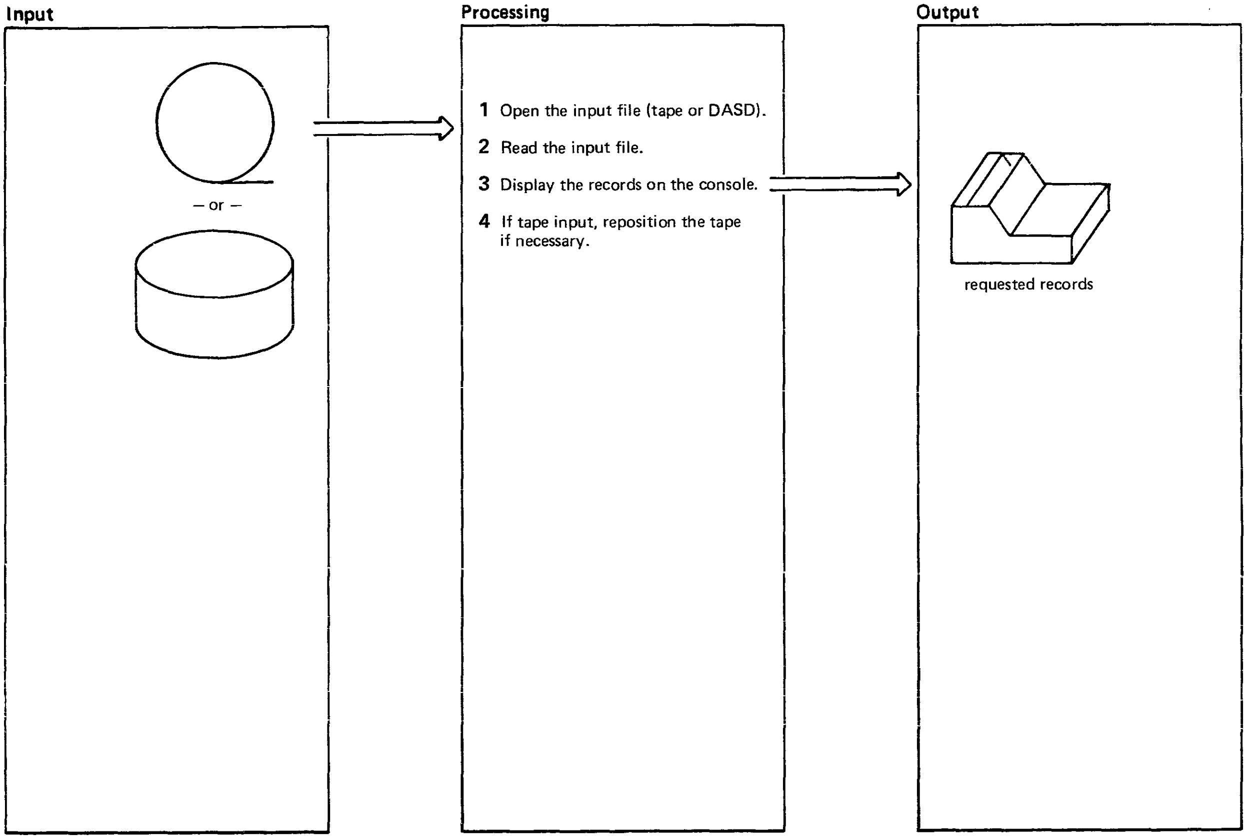
3 Print the records on the printer.

4 End the print function.

Notes	Module	Label	Ref
1 The input device is opened. If the input is on tape, the tape is spaced forward the designated number of records (if any). The extent table is updated to reflect the cylinders to be printed.	DMKDDR	OPENIN GETEXT	
2 The message PRINTING xxxxxxxx is displayed to indicate the start of the PRINT function.	DMKDDR	MSG004	
3 The data is read from the input device via the appropriate (disk or tape) read routine. The data is converted and printed on the system printer.	DMKDDR	BUILDTHR GETTHR DISPLAY	
4 The message END OF PRINT indicates the successful completion of the PRINT function.	DMKDDR	EOJ	
5 Control returns to the GTCARD routine to read the next control statement.			

Notes	Module	Label	Ref

Diagram 5-6. The Print Function



1 Open the input file (tape or DASD).

2 Read the input file.

3 Display the records on the console.

4 If tape input, reposition the tape if necessary.

Notes	Module	Label	Ref
1 The input device (either tape or disk) is opened. If input is on tape, the tape is spaced forward the designated number of records (if any). The extent table is updated to reflect the data to be typed.	DMKDDR	OPENIN GETEXT	
2 The records are read from the tape or disk by the appropriate read routine.	DMKDDR	BUILDTHR GETTHR	
3 The records are displayed on the console. The read and type cycle is continued until all the specified records are typed.	DMKDDR	DISPLAY	
4 Control returns to the GTCARD routine to read the next control statement.	DMKDDR	EOJ	

Notes	Module	Label	Ref

Diagram 5-7. The Type Function

Program Organization

This section contains a program description of the DMKDDR module.

DMKDDR

The DASD dump restore program.

Attributes
Serially reusable.

Entry Point
DMKDDREP.

Registers at entry
R1: Points to a parameter list when DMKDDR is executed under the control of CMS.

Registers at exit
R15: Contains a return code when DMKDDR is executed under the control of CMS. The return codes are:

Return Code	Meaning
1	Invalid filename or file not found.
2	Error while running the program.
3	Flagged DASD track.
4	Permanent tape or DASD I/O error.
1xx	Error in the PRINTIO routine.
2xx	Error in the CONREAD routine.
3xx	Error in the RDBUF routine.
4xx	Error in the TYPLIN routine.

where:

xx is the return code from the CMS routine.

Register Usage

R0: Work Register.

R1: Pointer to input field from SCANCONT.
Pointer to the output buffer (PRINT/TYPE).
Work register.

R2: Input count from SCANCONT.
Unit address for STARTIO.
Data block count (PRINT/TYPE).
Work register.

R3: End of current line (PRINT/TYPE).
Work register.

R4: Length of one line (PRINT/TYPE).
Pointer to key (PRINT/TYPE).
Work register.

R5: Total length of data (PRINT/TYPE).
Work register.

R6: Data count (PRINT/TYPE).
Number of records on the track (PRINT/TYPE).
Work register.

R7: Pointer to the extent table entry.
Current line pointer (PRINT/TYPE).

R8: Extent table entry size.
Last line pointer (PRINT/TYPE).

R9: Base register 5.

R10: Base register 1.

R11: Base register 2.

R12: Base register 3.

R13: Base register 4.

R14: Return address

R15: Pointer to the IOB.

External References
DMSACF, DMSCRD, DMSCWR

Directory

Figure 5-2 is an alphabetic list of the major labels in the DASD Dump Restore program. The associated method of operation diagrams are indicated and a brief description is included of the operation performed at the point in the program that is associated with each label.

Label	Diagram	Description
ADDLINE		Checks for duplicate line.
BINCONV		Converts decimal numbers to binary.
BUILDCCW		Builds a CCW string to put the key/data fields into the THR (track header record).
BUILDTHR	5-3 5-6 5-7	Reads records from disk.
CLOSEJOB	5-3 5-4 5-5	Displays message indicating the end of a DDR function.
CLOSE1		Closes the tape and reads another.
CMS1	5-1	Builds a PLIST (parameter list) if parameters passed from CMS.
CMS8	5-1	The end-of-job processing when DDR is running under VM/370.
COMPARE		Compares keywords.
DASDWRIT	5-4	Writes records onto disk.
DDR709		Issues DMKDDR709E message.
DDR714		Issues DMKDDR714E message.
DDR721		Issues DMKDDR721E message.
DDR724		Issues DMKDDR724E message.
DECCONV		Converts decimal numbers to hexadecimal.
DISPIT		Displays the key/data message.
DISPLAY	5-6 5-7	Prints or types records.
DMKDDR		Start of the DMKDDR module.
DMKDDREP	5-1	Entry point to the DDR program.
EOJ	5-3 5-6 5-7	At the end of a DDR function, returns control to the GTCARD routine.
ERRCLOSE		Closes tape and reads alternate tape.
EXIT	5-1	Returns to CMS command environment or enters wait state at end of program.
GETEXT	5-3 5-4 5-5 5-6 5-7	Builds extent table.
GETR1		Checks for records that need to be printed.
GETTHR	5-4 5-6 5-7	Reads tape records.

Figure 5-2. The DASD Dump Restore Program Label Directory (Part 1 of 3)

Label	Diagram	Description
GOSUB1		Gets the next record.
GRAPHID		Handles I/O for display terminals.
GTCARD	5-1	Reads control cards.
HEXCONV		Converts hexadecimal numbers to decimal.
INOUTER		Handles tape and DASD errors.
LASTONE		Checks for last record.
LOOP12		Checks for last record to be displayed.
LOOP13		Determines the starting address.
MSGWRITE		Displays messages on the terminal.
MSG004	5-3 5-4 5-5 5-6	Prints message indicating start of Dump, Restore, Copy, or Print function.
NEWADD	5-1	Prints heading when DDR program running standalcne.
NEXTCYL		Updates pointer to next cylinder.
NEXTREC		Updates pointer to next record.
NEXTTCK		Updates pointer to next track.
NOSTART		Sets up starting address for DMKDDR721E message.
OK		Points to read CCWs to read THR.
OPENDASD		Opens a DASD.
OPENIN	5-3 5-4 5-5 5-6 5-7	Opens input devices.
OPENOUT	5-3 5-4 5-5	Opens output devices.
PBUFFER		Points to the print buffer.
PDATA		Sets up print pointer.
PRINTDAT		Prints the data.
PRINTH	5-3 5-4 5-5	Prints function heading.
PRINTER1		Updates the printer line count.
PRINTER2		Spaces the printer twice.
PRINT1		Checks that device type is console.
PRINT2		Displays message on console.
READCONT		Reads control statements.
READCT		Reads the home address, record 0, and the count fields.
READKEYD		Reads the key and data records.
RETURN		Reorders the cylinder number.
SAVECT		Saves the printer line count.
SCANCONT		Scans control statements for next operand.
SCANCOPY	5-2	Scans the COPY function statement.
SCANDUMP	5-2	Scans the DUMP function statement.
SCANINPU	5-2	Scans the INPUT control statement.
SCANNAME		Scans the name table (TABLE1) for a matching control statement name.
SCANOUTP	5-2	Scans the OUTPUT control statement.
SCANPRIN	5-2	Scans the PRINT function statement.
SCANREST	5-2	Scans the RESTORE function statement.
SCANSYSP	5-2	Scans the SYSPRINT control statement.
SCANTYPE	5-2	Scans the TYPE function statement.

Figure 5-2. The DASD Dump Restore Program Label Directory (Part 2 of 3)

Label	Diagram	Description
SCANUNIT		Scans the device table (TAELE2).
SETDASD	5-4	Checks volume serial number of output disk.
SETEND		Prints the cylinder map at end-of-job.
SETEXT		Picks up the cylinder number that starts the next extent.
SETUPBUF		Clears the print buffer.
SKIPMSG		Prints record overflow message.
STARTIO		Starts I/O devices.
SUPMSG		Prints the suppress line message.
TESTCARD		Checks for card input at end-of-job.
TESTCMS	5-1	Exits by entering wait state when DDR program is running standalone.
TESTEND		Terminates when blank card read.
TESTIN		Checks for tape input.
TESTNPAG		Skips printer to channel 1.
TESTOUT	5-3	Writes tape output records.
TRANS		Translates data to printatle characters.
TRKCOND		Recovery procedure for track condition check (alternate track).
TSTCOUNT		Prints the end of the track.
UPDTADD	5-3 5-4 5-5	Updates disk addresses.
UPDTEXT		Restores entire track.
WDSIO		Writes the THR (track header record).

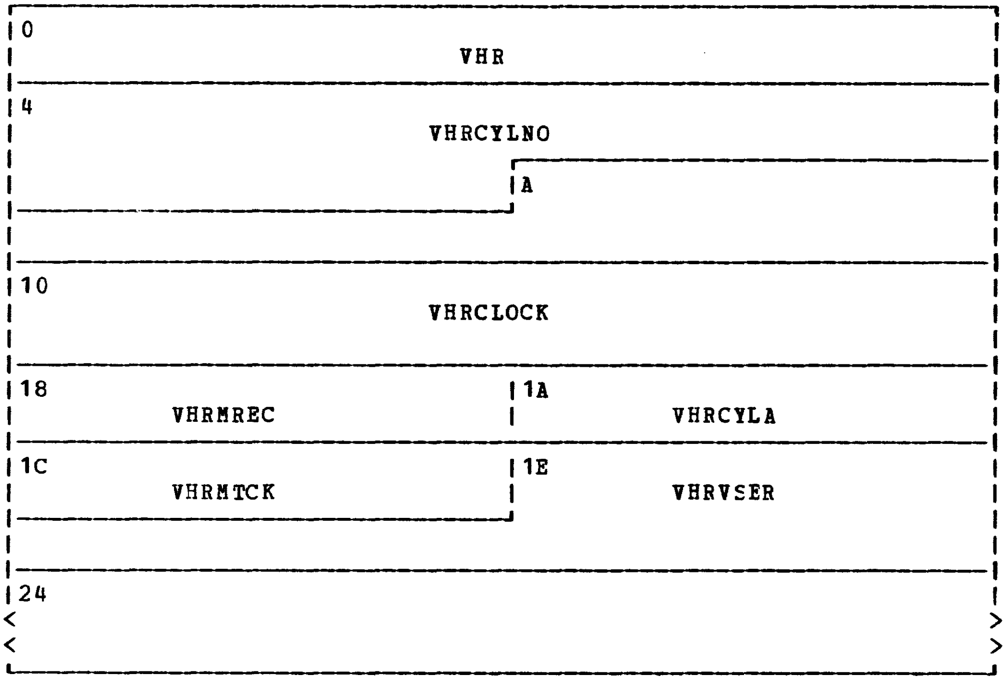
Figure 5-2. The DASD Dump Restore Program Label Directory (Part 3 of 3)

Data Areas

This section contains a description of a:

- Track header record
- Cylinder header record
- IOB

CYLINDER HEADER RECORD



Displacement					
Hex	Dec	Field Name			Description
0	0	VHR	DC	CL4'VHR'	
4	4	VHRCYLNO	DS	CL6'0'	BBCCHH of input DASD unit
A	10		DS	XL6' '	Not used
10	16	VHRCLOCK	DS	D'0'	Time of day clock value
18	24	VHRMREC	DS	X'0'	
1A	26	VHRCYLA	DS	H'0'	CC address of last cylinder on this type of DASD.
1C	28	VHRMTCK	DS	H'0'	
1E	30	VHRVSER	DS	CL6'VOLSER'	Volume serial number of input DASD unit
24	36		DS	CL44' '	

Figure 5-3. Cylinder Header Record

TRACK HEADER RECORD

```
┌────────────────────────────────────────────────────────────────┐
|0                                                               |
|                            THR                                 |
|────────────────────────────────────────────────────────────────|
|4                                |6                             |
|          THRNDRD                |          THRNDRT             |
|────────────────────────────────────────────────────────────────|
|8                                |A              |B             |
|          THRNDRL                |    THRFLAG    |              |
|─────────────────────────────────────────────────┘              |
|                                                                |
|                           THRHADD                              |
|────────────────────────────────────────────────────────────────|
|10                                                              |
|                                                                |
|                                                                |
|                           THR000                               |
|                                                                |
|                                                                |
|                                                                |
|                                                                |
|────────────────────────────────────────────────────────────────|
|20                                                              |
|                                                                |
|                           THR001                               |
|                                                                |
|────────────────────────────────────────────────────────────────|
<28 to FFF                                                       >
<                  (Count Field or Data)                         >
└────────────────────────────────────────────────────────────────┘
```

Displacement					
Hex	Dec	Field Name			Description
0	0	THR	DC	CL4'THR '	ID of track header record
4	4	THRNDRD	DC	H'0'	The number of count fields in the THR
6	6	THRNDRT	DC	H'0'	The number of 4K data records on tape
8	8	THRDRL	DC	H'0'	Length of the short (last) data record
A	10	THRFLAG	DC	XL1'0'	Flag
		Bit settings for THRFLAG			
		SPECIAL	EQU	X'01'	Overflow
B	11	THRHADD	DC	XL5'0'	The home address reordered
10	16	THR000	DC	XL16'0'	Record 0 from the DASD unit
20	32	THR001	DC	XL8'0'	Count field of the first record
28	40				Count fields and data

Figure 5-4. Track Header Record

```
┌─────────────────────────────────────────────────────────────────────┐
|0              |1                 |2                                 |
|   IOBSTAT     |     IOBOPT       |          IOBUADD                 |
|─────────────────────────────────────────────────────────────────────|
|4                                                                    |
|                               IOBCCW                                |
|─────────────────────────────────────────────────────────────────────|
|8                                                                    |
|                              IOBERROR                               |
|─────────────────────────────────────────────────────────────────────|
|C                                                                    |
|                                                                     |
|                               IOBCSW                                |
|                                                                     |
|─────────────────────────────────────────────────────────────────────|
|14             |15                |16                                |
|  IOBCLASS     |     IOBTYPE      |           IOBMREC                |
|─────────────────────────────────────────────────────────────────────|
|18                                |1A                                |
|          IOBCYLP                 |             IOBCYLA              |
|─────────────────────────────────────────────────────────────────────|
|1C                                |1E               |1F              |
|          IOBMTCK                 |    IOBMODE      |    IOBDISP     |
|─────────────────────────────────────────────────────────────────────|
|20                                                                   |
|            IOBVSER               ┌──────────────────────────────────|
|24                                |26                                |
|                                  |            IOBATAPE              |
|─────────────────────────────────────────────────────────────────────|
|28             |29                                                   |
|  IOBFLAG      |             Reserved for IBM use                    |
└─────────────────────────────────────────────────────────────────────┘
```

Displacement					
Hex	Dec	Field Name			Description
0	0	IOBSTAT	DS	X'80'	Status of IOB
		Bit settings for IOBSTAT			
		IOBST	EQU	X'80'	I/O unit is to be started
		IOBSTACK	EQU	X'40'	I/O error has been stacked
		IOBLAST	EQU	X'20'	Last IOB
		IOBNOPER	EQU	X'10'	Device is not operational
		IOBCPVOL	EQU	X'08'	Unit is a CPVOL
		IOBOPEN	EQU	X'04'	The IOB is open
		IOBSCRAT	EQU	X'02'	The DASD device is a scratch volume
		IOBTPSWP	EQU	X'01'	Switch to alternate tape in progress
1	1	IOBOPT	DS	1X	IOB flags
		Bit settings for IOBOPT			
		IOBDEW	EQU	X'80'	Wait for device end interrupt
		IOBERST	EQU	X'40'	Stop on I/O error and wait for next interrupt
		IOBEEXIT	EQU	X'20'	Repeat CCW on error
		IOBSIO	EQU	X'10'	Do not use Diagnose I/O

Figure 5-5. IOB (Input/Output Block) Format (Part 1 of 2)

Displacement			
Hex	Dec	Field Name	Description
2	2	IOBUADD DS 1H	Unit address of device
4	4	IOBCCW DS 1F	Pointer to CCW
8	8	IOBERROR DS A	Address of IO error routine
C	12	IOBCSW DS 2F	CSW of IO error stacked
14	20	IOBCLASS DS X'0'	Device class
15	21	IOBTYPE DS X'0'	Device type
16	22	IOBSKIP EQU *	IOB type skip count
16	22	IOBMREC DS H'0'	Maximum number of records that will fit a track
18	24	IOBCYLP DS H'0'	Maximum primary cylinder address of DASD device.
1A	26	IOBCYLA DS H'0'	Maximum alternate cylinder address of DASD device.
1C	28	IOBMTCK DS H'0'	Maximum number of tracks (numbering C-N)
1E	30	IOBMODE DS X	IOB tape mode command code
1F	31	IOBDISP DS X	IOB tape disposition command code
20	32	IOBVSER DS CL6' '	Volume serial number of DASD unit
26	38	IOBATAPE DS X'0000'	Address of an alternate tape unit
28	40	IOBFLAG DS X'0'	IOB flag
29	41	DS 3X'0'	Reserved for IBM use
		IOBSIZE EQU *-IOB	Address of an alternate tape unit

Figure 5-5. IOB (Input/Output Block) Format (Part 2 of 2)

Diagnostic Aids

Figure 5-6 lists the messages issued by the DASD Dump Restore Program. The associated label and method of operation diagram are included in the list.

Message Code	Label	Diagram	Message Text
DMKDDR700E	DDR700		INPUT UNIT IS NOT A CPVOL
DMKDDR701E	DDR701	5-2	INVALID OPERAND - xxxxxxxxxxxx
DMKDDR702E	DDR702		CONTROL STATEMENT SEQUENCE ERROR
DMKDDR703E	DDR703	5-2	OPERAND MISSING
DMKDDP704E	DDR704		DEV ccu NOT OPERATIONAL
DMKDDR705B	DDR705		IO ERROR ccu SENSE xxxxxxxxxxxx CSW xxxxxxxxxxxxxxxx
DMKDDR707E	DDR707		MACHINE CHECK RUN SEREP AND SAVE OUTPUT FOR CE
DMKDDR708E	DDR708		INVALID INPUT OR OUTPUT DEFINITION
DMKDDR709E	DDR709		WRONG INPUT TAPE MOUNTED
DMKDDR710A	DDR710		DEV ccu INTERVENTION required.
DMKDDR711R	DDR711		VOLID READ IS volse1 NOT volse2
DMKDDR712E	DDR712		NUMBER OF EXTENTS EXCEEDS 20
DMKDDR713E	DDR713		OVERLAPPING OR INVALID EXTENTS
DMKDDR714E	DDR714		RECORD xxxxxxxxxxxx NOT FOUND ON INPUT TAPE
DMKDDR715E	DDR715		LOCATION xxxxxxxxxxxx IS A FLAGGED TRACK
DMKDDR716R	DDR716		NO VOL1 LABEL FOUND FOR xxxxxx
DMKDDR717R	DDR717	5-4	DATA DUMPED FROM volse1 TO BE RESTORED TO volse2
DMKDDR718E	DDR718		OUTPUT UNIT IS FILE PROTECTED
DMKDDR719E	DDR719		INVALID FILENAME OR FILE NOT FOUND
DMKDDR720E	DDR720		ERROR IN xxxxxxxx

Figure 5-6. The DASD Dump Restore Program Messages (Part 1 of 2)

Message Code	Label	Diagram	Message Text
DMKDDR721E	DDR721		RECORD xxxxxxxxxxxx NOT FOUND.
DMKDDR722E	DDR722		OUTPUT UNIT NOT PROPERLY FORMATTED FOR THE CP NUCLEUS
DMKDDR723E	DDR723		NO VALID CP NUCLEUS ON THE INPUT UNIT
DMKDDR724E	DDR724		INPUT TAPE CONTAINS A CP NUCLEUS DUMP
DMKDDR725R	DDR725	5-2,5-4	ORIGINAL INPUT DEVICE WAS (IS) LARGER THAN OUTPUT DEVICE
DMKDDR726E	DDR726		MOVING DATA INTO THE ALTERNATE TRACK CYLINDER(S) IS PROHIBITED.
DMKDDR727E	DDR727		FLAGGED TRK xxxxxxxxxxxx HAS NO PROPER ALTERNATE; SKIPPING THIS TRK.
DMKDDR756E	DDR706		PROGRAM CHECK PSW = xxxxxxxxxxxxxxxx
	MSG002	5-1	VM/370 DASD DUMP/RESTORE PROGRAM RELEASE n
	NEWADD		
	MSG02A		ENTER CARD READER ADDRESS OR CONTROL STATEMENTS
	MSG003		ENTER CYLINDER EXTENTS
	MSG03B		ENTER NEXT EXTENT OR NULL LINE
	MSG005		END OF VOLUME CYL xxx HD xxx, MOUNT NEXT TAPE
	MSG004		RESTORING xxxxxx
	MSG004		COPYING xxxxxx
	MSG004		DUMPING xxxxxx
	MSG004	5-6	PRINTING xxxxxx
	MSG001	5-3	END OF DUMP
	CLOSEJOB	5-3	END OF RESTORE
		5-3	END OF COPY
	MSG001	5-3	END OF PRINT
	EOJ		
	MSG001		END OF JOB
	RESPMSG		DO YOU WISH TO CONTINUE? RESPOND YES NO OR REREAD:
	RESPMSG2		DO YOU WISH TO CONTINUE? RESPOND YES OR NO

Figure 5-6. The DASD Dump Restore Program Messages (Part 2 of 2)

Chapter 6. The Installation Verification Procedure

Introduction

The Installation Verification Procedure (IVP) for VM/370 is designed to exercise the generated system to verify that basic VM/370 facilities are operable. The IVP is contained in two files using the EXEC facility of CMS, and uses two virtual machines in addition to the system operator's virtual machine.

The tests exercise the following areas of CP:

- Multiple virtual machine support
- I/O spooling
- Transferring of spooled data to other virtual machines
- Offline I/O operations
- Sending of messages to the system operator
- Paging operations
- Task dispatching and scheduling
- Disk I/O support
- Automatic warm start following abnormal termination of VM/370

The following facilities of CMS are exercised:

- Normal CMS command processing
- Disk formatting
- Copying of files
- Creation and modification of files via EDIT command
- Assembly of executable programs
- Execution of user programs
- Creation and execution of user-written commands
- Printing and punching of CMS files
- Issuing of commands to CP
- Use of multilevel nested EXEC procedures
- Stacking and unstacking of command and data input from the terminal
- Communication with user from EXEC procedures

Several other system facilities, incidental to the primary IVP tests, are exercised. Certain system facilities, such as preferred execution options, virtual=real, OS ISAM, and VSAM and Access Method Services under CMS, are not exercised by the IVP.

The IVP requires operator intervention only when an operational decision is to be made, or to initiate the IVP tests themselves. All file creation, erasure, management, and logoff of the virtual machines (with the exception of the system operator) at test completion is performed automatically without operator or user action.

Method of Operation

This section describes the execution of the two EXEC procedures of the IVP (Installation Verification Procedure).

Figure 6-1 shows the relationship of the diagrams.

Diagram 6-1 describes the highest level EXEC procedure, IVP.

Diagram 6-2 describes the major functions of the nested EXEC procedure IVPX.

Diagram 6-3 describes test procedure 1.

Diagram 6-4 describes test procedure 2.

Diagram 6-5 describes the error processing.

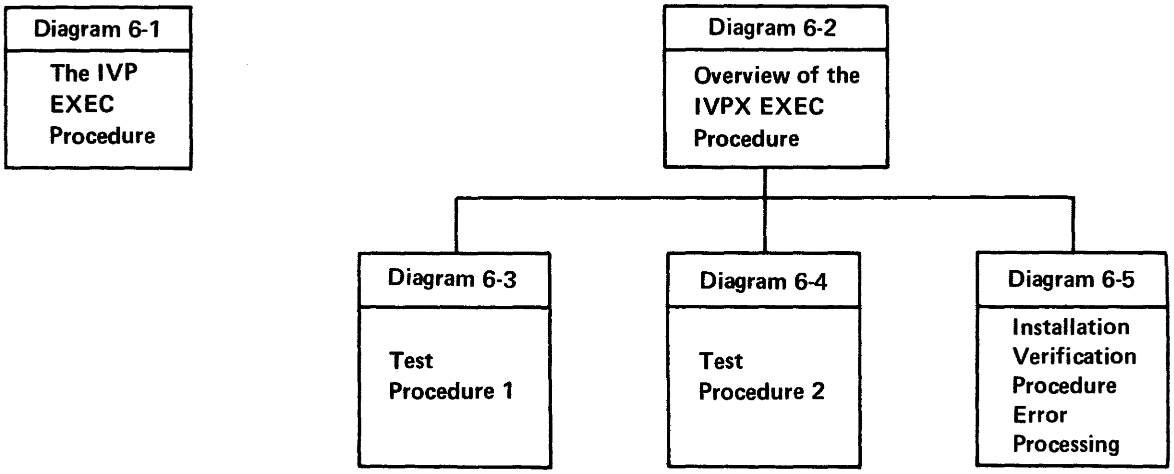
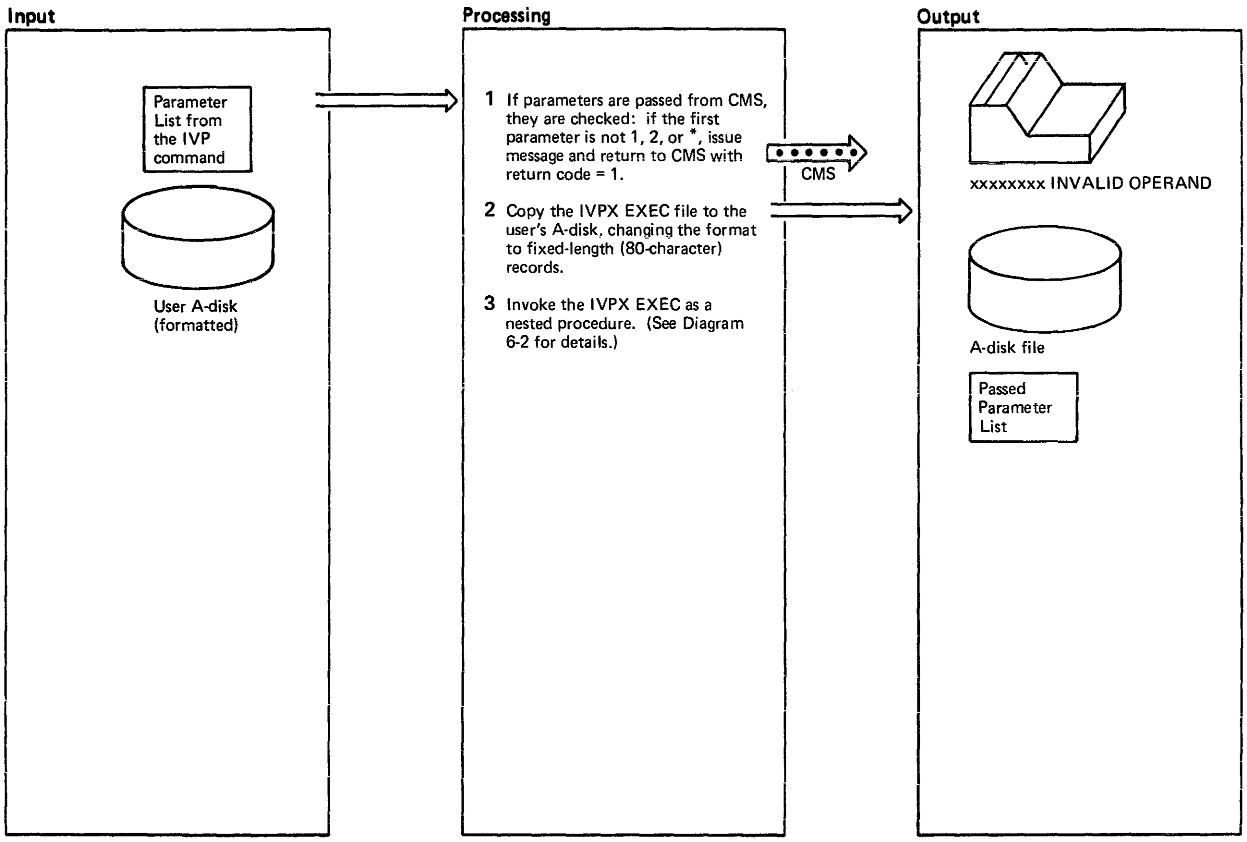


Figure 6-1. Key to the Installation Verification Procedure Method of Operation Diagrams

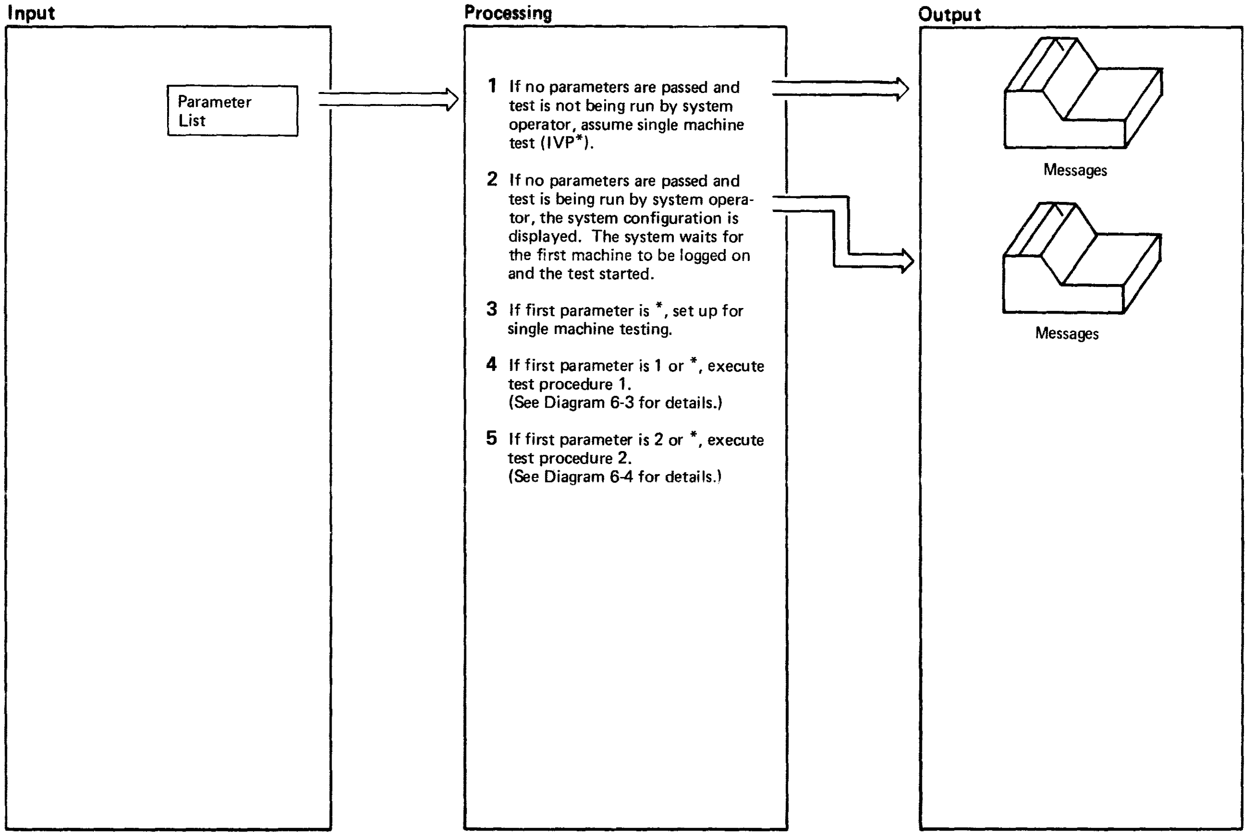


1 If parameters are passed from CMS, they are checked: if the first parameter is not 1, 2, or *, issue message and return to CMS with return code = 1.

2 Copy the IVPX EXEC file to the user's A-disk, changing the format to fixed-length (80-character) records.

3 Invoke the IVPX EXEC as a nested procedure. (See Diagram 6-2 for details.)

Diagram 6-1. The IVP EXEC Procedure



1 If no parameters are passed and test is not being run by system operator, assume single machine test (IVP*).

2 If no parameters are passed and test is being run by system operator, the system configuration is displayed. The system waits for the first machine to be logged on and the test started.

3 If first parameter is *, set up for single machine testing.

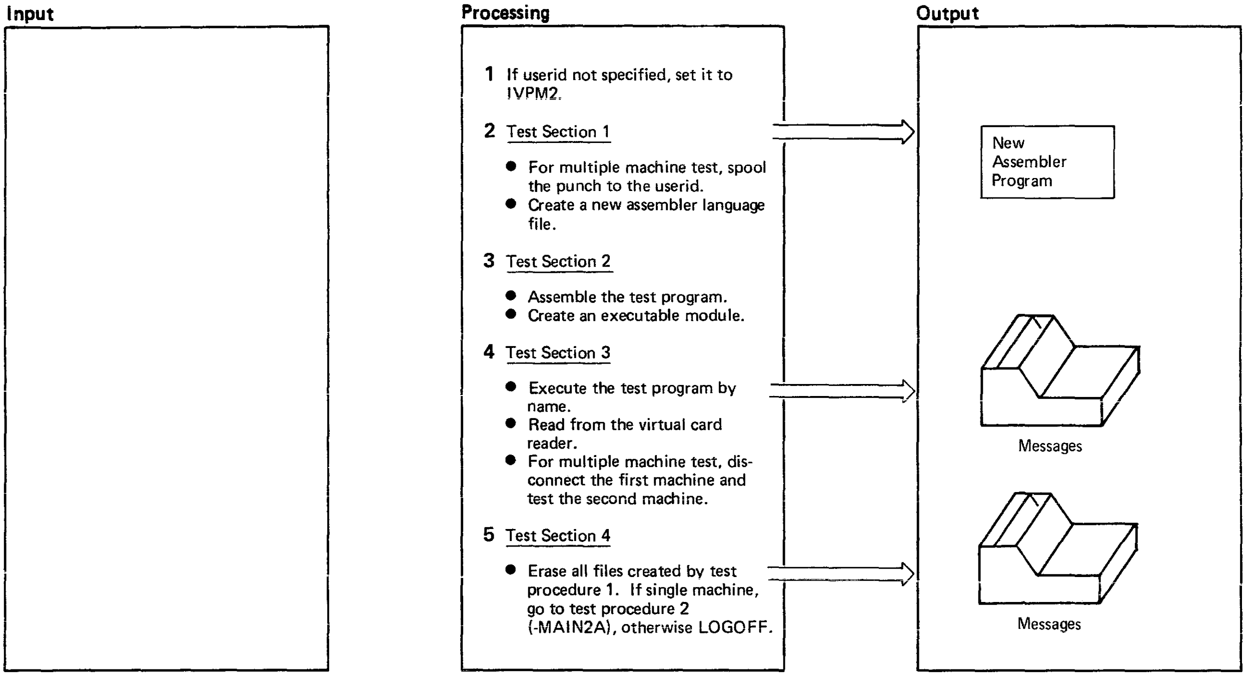
4 If first parameter is 1 or *, execute test procedure 1. (See Diagram 6-3 for details.)

5 If first parameter is 2 or *, execute test procedure 2. (See Diagram 6-4 for details.)

Notes	Module	Label	Ref
1 When no parameters are specified on the IVP command, the message *** ARE YOU THE SYSTEM OPERATOR? ENTER "YES" OR "NO" is displayed. If the response is NO, the message *** NOT SYSTEM OPERATOR-DEFAULT TO IVP * is displayed, single machine testing is set up [-INIT], and the testing starts at test procedure 1.	IVPX	-CKOP	
2 The real system configuration is displayed. The messages *** FROM A TERMINAL, ENTER THE FOLLOWING FOUR COMMANDS LOGIN IVPM1 (WHEN REQUESTED, ENTER THE PASSWORD IVPASS) DEFINE STORAGE AS 16,384K IPL 190 IVP 1 give instructions for running the standard test procedure 1. Then the virtual machine enters a dormant state which can be interrupted by signalling attention from the terminal. The message *** THIS PORTION OF IVP NOW GOING TO SLEEP is displayed and the system waits.	IVPX	-CKOP	

Notes	Module	Label	Ref
3 Set &GLOBAL2=4 to indicate single machine test. Erase all CMS files with filenames IVPTST and IVPTST2. If return code is other than 0 or 2, the ERASE command (to erase the EXEC file) is stacked in the terminal and control returns to the CMS command environment. If the return code is 0 or 2, test procedure 1 (MAIN1A) is executed.		-INITB -GETOUT	

Diagram 6-2. Overview of the IVPX EXEC Procedure



1 If userid not specified, set it to IVPM2.

2 Test Section 1

- For multiple machine test, spool the punch to the userid.
- Create a new assembler language file.

3 Test Section 2

- Assemble the test program.
- Create an executable module.

4 Test Section 3

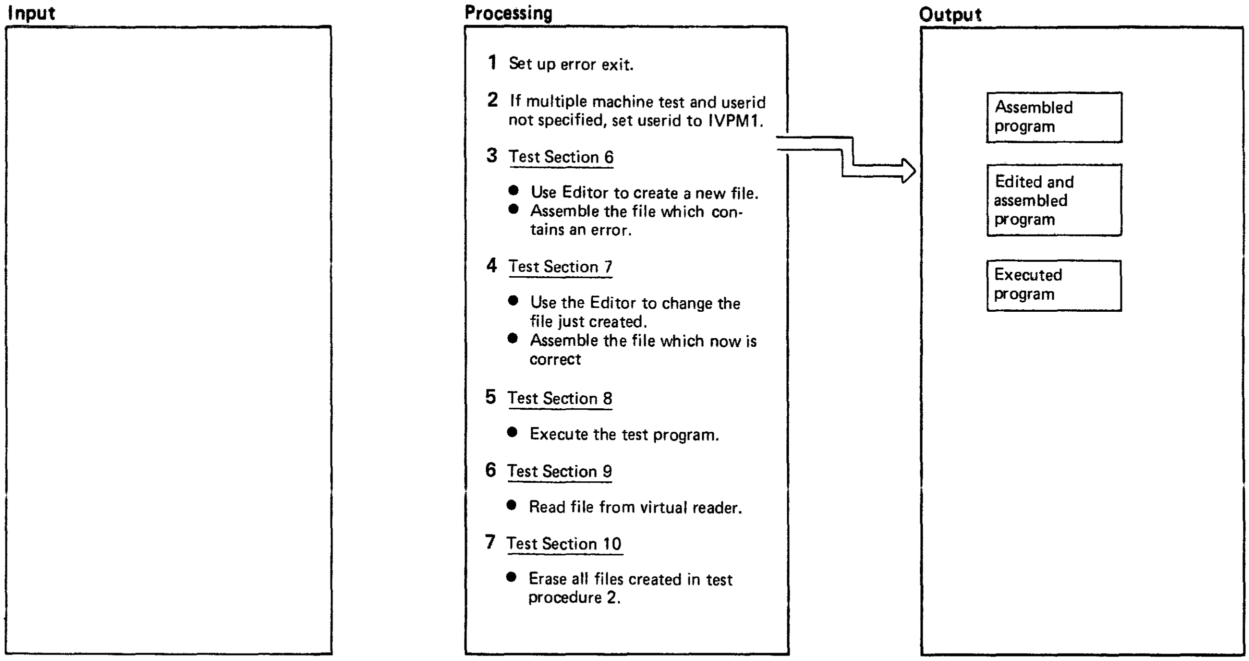
- Execute the test program by name.
- Read from the virtual card reader.
- For multiple machine test, disconnect the first machine and test the second machine.

5 Test Section 4

- Erase all files created by test procedure 1. If single machine, go to test procedure 2 (-MAIN2A), otherwise LOGOFF.

Notes	Module	Label	Ref
1 For a multiple machine test, the userid is set to IVPM2 or to the userid specified as the second operand of the IVP command. When the userid is set to IVPM2, &GLOBAL5 is set to 2 to indicate the standard test.	IVPX	-MAIN1	
2 The assembler language statements are stacked in the terminal input buffer and edited.	IVPX	-MAIN1A	
3 The test program created in test section 1 is first assembled (ASSEMBLE command) and then made executable by issuing the LOAD and GENMOD commands.	IVPX	-K256	
4 The test program, IVPTST, is executed. Next a READ is issued to the virtual reader and a return code is requested.	IVPX	-LOOPA	
If the return code is other than 0 or 8, the ERASE command to erase the EXEC file is stacked in the terminal, and control returns to the CMS command environment.		-GETOUT	
When testing multiple machines, the following messages are issued: *** WHEN "VM/370 ONLINE" APPEARS, ENTER THE FOLLOWING THREE COMMANDS LOGIN userid (WHEN REQUESTED ENTER THE APPROPRIATE PASSWORD) (IF LOGGING IN IVPM2, THE PASSWORD IS: IVPASS) IPL 190 IVP 2 *** THIS PORTION OF IVP NOW DISCONNECTING			
The first machine is then disconnected. The operator enters the above commands to start the second machine. The procedure loops (control keeps returning to -LOOPA) until the file to start the second machine is spooled to the reader. The STATE command is issued to verify the existence of the file. The second machine is started.		-FINIS	
5 All the IVPTST files are erased. If the test machine is still connected (&GLOBAL2≠3) the following messages are issued. *** TEST SECTION 5 RESERVED FOR FUTURE USE *** *** IVP TEST 1 SUCCESSFULLY COMPLETED These same messages are sent to the punch if the test machine is already disconnected (&GLOBAL=3). The single machine test resumes at -MAIN2A, test procedure 2. If the standard test is running the message *** IVP TEST 1 FINISHED	IVPX	-INLINE	
is sent to the system operator. If &GLOBAL5=1, the test is running in 256K bytes of storage. If running machine tests, go to the LOGOUT routine. The following commands are stacked. ERASE IVPX EXEC A1 CP LOGOUT The LOGOUT routine closes all files including the punch containing the messages issued after test machine 1 was disconnected. The multiple machine test resumes at -MAIN2, test procedure 2.		-LOGOUT	

Diagram 6-3. Test Procedure 1



Input

Processing

1 Set up error exit.

2 If multiple machine test and userid not specified, set userid to IVPM1.

3 Test Section 6

- Use Editor to create a new file.
- Assemble the file which contains an error.

4 Test Section 7

- Use the Editor to change the file just created.
- Assemble the file which now is correct

5 Test Section 8

- Execute the test program.

6 Test Section 9

- Read file from virtual reader.

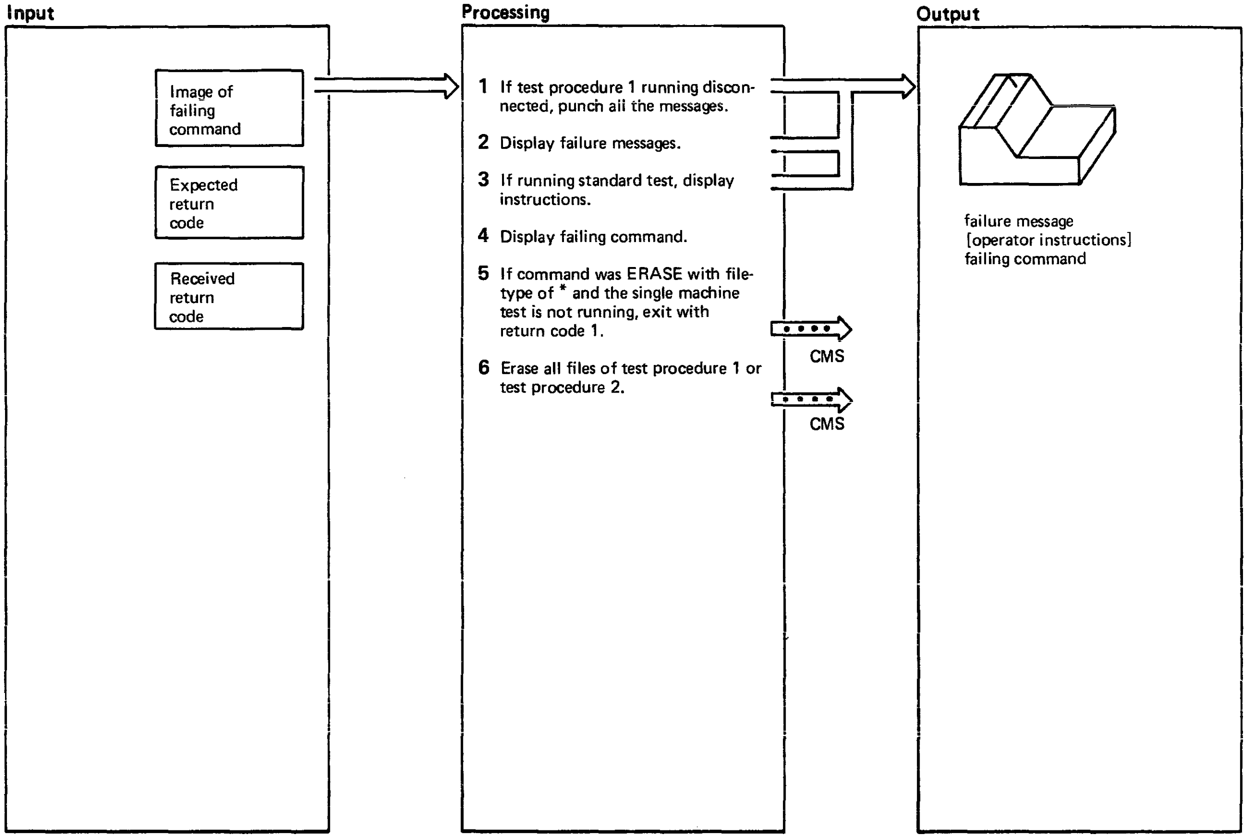
7 Test Section 10

- Erase all files created in test procedure 2.

Output

Notes	Module	Label	Ref
1 Set the error exit to -FAIL2. For a single machine test, exit directly to the CMS command environment. Otherwise, display the instruction *** WHEN "VM/370 ONLINE" APPEARS, ENTER THE FOLLOWING TWO COMMANDS: LOGIN xxxxxxxx (WHEN REQUESTED, ENTER THE APPROPRIATE PASSWORD) LOGOUT The ERASE and LOGOUT commands are stacked in the terminal and the EXEC procedure exits with a return code of 1. Execution is now ended within the nested EXEC. The return code of 1 forces the next level EXEC to exit to the CMS command environment.	IVPX	-FAIL2	
2 For a multiple machine test, the userid is set to IVPM1 or to the userid specified as the second operand of the IVP command. When the userid is set to IVPM1, &GLOGAL5 is set to 2 to indicate the standard test.	IVPX	-MAIN2	
3 The input data is stacked for the editor, which creates the IVPTST2 ASSEMBLE file. The file just created is assembled. Error 8 occurs because the ASSEMBLE file contains one error.	IVPX	-MAIN2A	
4 The statement in error is corrected. The file is then assembled. Since the error is corrected the TEXT file is created.	IVPX		
5 The test program is loaded and then started.	IVPX	-MAIN2A	
6 The file is read from the virtual reader. If there is no file in the reader on the first loop, a file is created, punched, and spooled to the reader.	IVPX	-LOOP -LOOP2	
For a single machine test, a dummy message file is created, punched, and spooled to the reader on the same machine. For a multiple machine test, the messages are spooled to the reader on the userid system. The input is stacked in the terminal for the editor. A dummy message is edited and punched. Control returns to -LOOP.			
The STATE command is issued to be sure the file is successfully read onto disk. The contents of the file are displayed. For multiple machine standard test, the message DON'T START SPOOL DEVICES UNTIL TOLD is sent to the system operator. The multiple machine test determines that the file was successfully read and punches and prints that file.		-LOOP1	
7 All files are erased and messages are displayed. *** IVP TEST 2 SUCCESSFULLY COMPLETED *** IVP PROCEDURE FINISHED	IVPX	-NOSPL	
If a single machine test, the command to erase the EXEC file is stacked in the terminal and control returns to the CMS command environment. If a multiple machine test, the commands to erase the EXEC file and LOGOUT are stacked for CMS. If running the standard test, the messages *** IVP TEST NOW FINISHED *** SIGNAL ATTN AND ENTER: BEGIN are sent to the system operator. For the multiple machine test, control then returns to the CMS command environment.		-GETOUT	

Diagram 6-4. Test Procedure 2



Input

Image of failing command

Expected return code

Received return code

Processing

1 If test procedure 1 running disconnected, punch all the messages.

2 Display failure messages.

3 If running standard test, display instructions.

4 Display failing command.

5 If command was ERASE with filetype of * and the single machine test is not running, exit with return code 1.

6 Erase all files of test procedure 1 or test procedure 2.

Output

failure message
[operator instructions]
failing command

Notes	Module	Label	Ref
1 If test machine 1 is disconnected the messages are sent to the punch, rather than the virtual machine console.	IVPX	-CHECK1	
2 The message *** IVP FAILURE HAS OCCURRED *** is displayed.	IVPX		
3 The messages *** IVP HAS FAILED – REPLY NO TO ABORT MESSAGE *** SIGNAL ATTN AND ENTER: BEGIN are sent to the system operator.	IVPX		
4 The messages *** COMMAND: xxxxxxxx *** EXPECTED RETURN CODE xxx *** RECEIVED RETURN CODE xxx are displayed.	IVPX	-CHECK2	
5 Control returns to the next level EXEC procedure and the return code of 1 forces that level to return to the CMS command environment.	IVPX		
6 If the number of the test section is less than 6, all the IVPTST files are erased. If the number of the test section is greater than 5, all the IVPTST2 files are erased. Because this is a nested EXEC procedure, exit with a nonzero return code. A nonzero return code forces the next level EXEC to return to the CMS command environment.	IVPX	-QUIT	

Diagram 6-5. Installation Verification Procedure Error Processing

Program Organization

The IVP (Installation Verification Procedure) consists of two EXEC procedures: IVP and IVPX. Figure 6-2 shows the structuring of the major routines of the IVP. Figure 6-3 relates the test sections to the CP or CMS functions being exercised.

INSTALLATION VERIFICATION PROCEDURE ROUTINE STRUCTURING

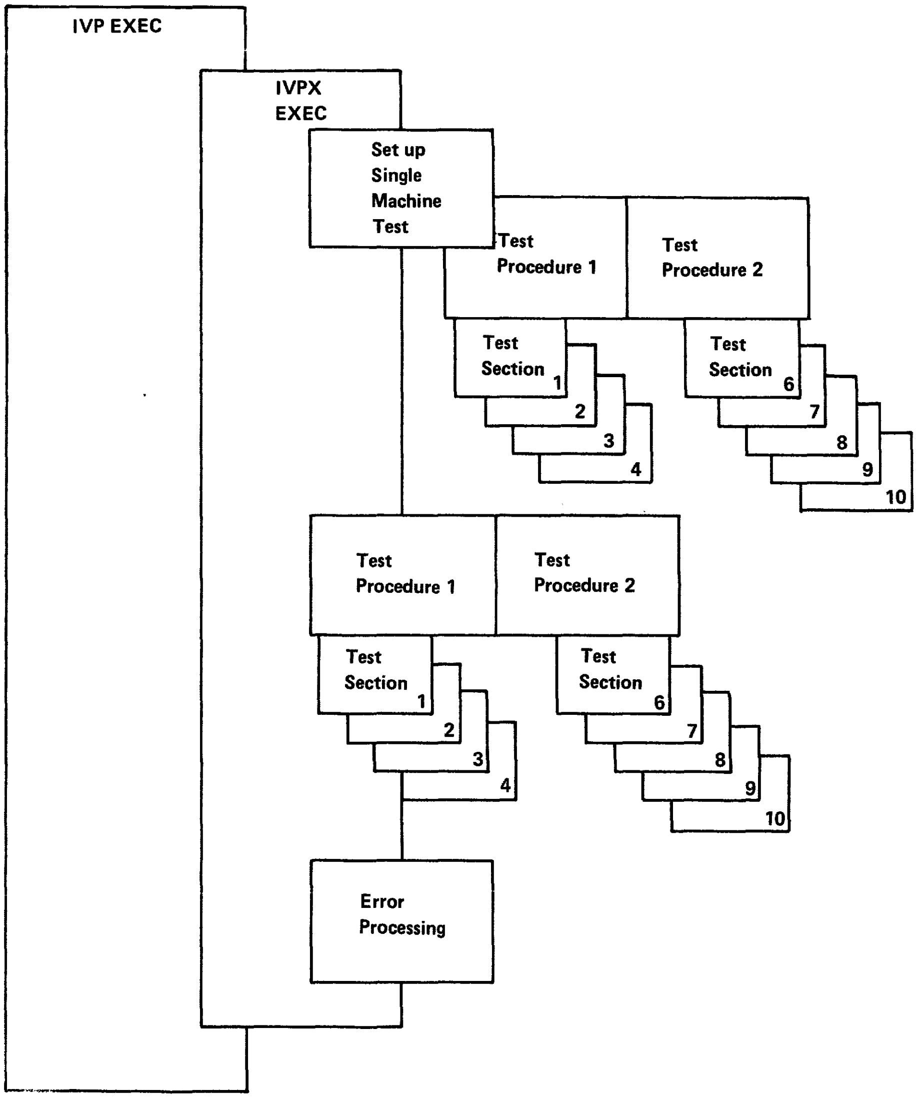


Figure 6-2. Structure of Installation Verification Procedure Routines

Program	Function Tested	Test Section and Comments
CP	Multiple virtual machine support	Test Procedures 1 and 2 test multiple virtual machine support when IVP * is not specified or assumed.
	I/O Spooling	Test Section 9.
	Transferring of spooled data to other virtual machines	Test Section 9 when IVP * is not specified or assumed.
	Offline operations	Test Section 9.
	Sending messages to system operator.	Test Sections 4 and 9.
	Page operations	Used throughout IVP.
	Task dispatching and scheduling	Used throughout IVP.
	Disk I/O support	Used throughout IVP.
	Automatic warm start	Error processing.
CMS	Command processing	Used throughout IVP.
	Copying of files	The IVP EXEC procedure.
	Creation and modification of files via EDIT command	Test Sections 1, 6, and 7.
	Assembly of executable modules	Test Sections 2, 6, and 7.
	Execution of user programs	Test Sections 3 and 8.
	Creation and execution of user-written commands	Test Section 3.
	Printing and punching of CMS files.	Test Section 7.
	Multilevel EXEC procedures.	Used throughout IVP.

Figure 6-3. Installation Verification Procedure Tests

Directory

This section contains an alphabetical list of the labels in the IVPX EXEC procedure. Figure 6-4 describes the function performed at the point in the program corresponding to each label; the associated method of operation diagram is referenced.

Label	Diagram	Description
-CHECK1	6-5	Sends messages to punch when machine is disconnected.
-CHECK2	6-5	Displays the failing command.
-CKOP	6-2	Sets up for execution when IVP is invoked without any parameters specified.
-FAIL2	6-4	Exits to CMS command environment if single machine test is running. Issues instructions if multiple machine test is running.
-FINIS	6-3	End of Test Procedure 1.
-GETOUT	6-2 6-3 6-4	Error exit for single machine test.
-INITB	6-2	Sets up for single machine test.
-INLINE	6-3	Erases all files created during Test Procedure 1.
-K256	6-3	Assembles and executes the program created in Test Section 1.
-LOGOUT	6-3 6-4	Error exit for multiple machine test.
-LOOP	6-4	Reads file from the virtual reader during Test Procedure 2.
-LOOPA	6-3	Reads from the virtual reader during Test Procedure 1.
-LOOP1	6-4	Checks that file is read to disk successfully.
-LOOP2	6-4	Creates file, punches it, and spools it to reader when there is no file in the reader.
-MAIN1	6-3	Beginning of Test Procedure 1.
-MAIN1A	6-3	Point in Test Procedure 1 where the single machine test begins.
-MAIN2	6-4	Beginning of Test Procedure 2.
-MAIN2A	6-4	Point in Test Procedure 2 where the single machine test begins.
-NOSPL	6-4	Erases all files created in Test Procedure 2.
-QUIT	6-5	Abnormal end exit from a nested EXEC procedure.

Figure 6-4. Installation Verification Procedure Label Directory

Diagnostic Aids

Figure 6-5 is a list of all the messages that the IVPX EXEC procedure issues, the label nearest to the point where the message is issued, and the associated method of operation diagram.

Label	Diagram	Message Text
-CKOP	6-2	*** ARE YOU THE SYSTEM OPERATOR? ENTER "YES" OR "NO".
-CKOP	6-2	*** NOT SYSTEM OPERATOR - DEFAULT TO IVP *
-CKOP	6-2	*** FROM A TERMINAL, ENTER THE FOLLOWING FOUR COMMANDS: LOGIN IVPM1 (WHEN REQUESTED, ENTER THE PASSWORD IVPASS) DEFINE STORAGE AS 16384K IPL 190 IVP 1
-CKOP	6-2	*** THIS PORTION OF IVP NOW GOING TO SLEEP. *** STARTING SYSTEM ABORT ROUTINE.
-ABMSG		*** ENTER "GO" TO CONTINUE OR "NO" TO QUIT.
-ABMSG		*** THIS IS THE LAST STEP OF THE IVP PROCEDURE. *** FOLLOWING SYSTEM RESTART (WARM START), START SPOOLING DEVICES.
-ABMSG		MANUALLY DEPRESS CPU RESTART KEY TO ABORT SYSTEM.
-PERFORM		*** STARTING TEST SECTION x
-CHECK1	6-5	*** IVP FAILURE HAS OCCURRED ***
-CHECK1	6-5	*** IVP HAS FAILED - REPLY NO TO ABORT MESSAGE *** SIGNAL ATTN AND ENTER: BEGIN
-CHECK2	6-5	*** COMMAND: xxxxxxxx *** EXPECTED RETURN CODE xxx *** RECEIVED RETURN CODE xxx
-LOOPA	6-3	*** WHEN "VM/370 ONLINE" APPEARS, ENTER THE FOLLOWING THREE COMMANDS: LOGIN xxxxxxxx (WHEN REQUESTED, ENTER THE APPROPRIATE PASSWORD) (IF LOGGING IN IVPM2, THE PASSWORD IS: IVPASS) IPL 190 IVP 2 *** THIS PORTION IS NOW DISCONNECTING
-INLINE	6-3	*** TEST SECTION 5 RESERVED FOR FUTURE USE ***
-INLINE1	6-3	*** IVP TEST 1 SUCCESSFULLY COMPLETED *** IVP TEST 1 FINISHED
-LOOP1	6-4	DON'T START SPOOL DEVICES UNTIL TOLD.
-NOSPL	6-4	*** IVP TEST 2 SUCCESSFULLY COMPLETED *** IVP PROCEDURE FINISHED
-NOSPL	6-4	*** IVP TEST 2 FINISHED *** SIGNAL ATTN AND ENTER: BEGIN
-FAIL2	6-4	*** WHEN "VM/370 ONLINE" APPEARS, ENTER THE FOLLOWING TWO COMMANDS: LOGIN xxxxxxxx (WHEN REQUESTED, ENTER THE APPROPRIATE PASSWORD) LOGOUT

Figure 6-5. The Installation Verification Procedure Messages

Chapter 7. Procedures for Generating and Updating VM/370

Introduction

The VM/370 update facility provides for the updating of files with several levels of updates and any number of program temporary fixes (PTFs). For Assembler language source statement files, procedures are supplied for assembling the updated source code to produce a uniquely defined text deck. The deck has a unique name and some control cards to identify the origin of the updates, macro libraries, and source statements. For macro library files, a copy file is produced to identify the origin of the input and any updates applied.

Procedures are provided for generating load files from various object modules, and for generating MACLIB files from various COPY and MACRO files.

The procedure for updating VM/370 has a file naming convention for update and text files, a set of programs to support the processing, and a set of EXEC procedures and modules to process the files.

- The VMFASM procedure incorporates PTFs or updates.
- The GENERATE procedure generates a new standalone card deck on disk.
- The VMFLOAD module generates a new CP, CMS, or RSCS nucleus.
- The VMFMAC procedure generates a new macro library.

UPDATE FILES

Files used to update another file are given a filetype of UPDTxxxx, where xxxx is a unique _update identifier_ for programmer and system use. The filename of the update file must be the same name as the file to be updated. For instance, the file PROGRAM ASSEMBLE could be updated by the file PROGRAM UPDTGN30 or the file PROGRAM UPDTGC61.

The creation and use of update files are described in the UPDATE command discussion in the _VM/370: CMS Command and Macro Reference_.

TXT FILES

Text files are produced by the assembler as a part of the VMFASM procedure. The filename of the text file is the same as the filename of the ASSEMBLE file. The filetype of the completed text deck is TXTnamex, where 'namex' represents a unique _update level identifer_. The value of 'namex' is taken from a control file, and corresponds to the highest level of update applied. In addition, the text deck is produced from a combination of the assembler text deck and an auxiliary control file containing data describing the origin of the files used. The auxiliary file is called 'filename UPDATES' and is produced by a program called VMFDATE. The filename is the same as the filename of the UPDTxxxx file.

CONTROL FILES

Each user may have several control files to specify various combinations of updates and macro libraries to be used. A control file must have a filetype of CNTRL. These control files contain records in the following format:

```
_nam00_ MACS maclib1 maclib2 ...
_nam01_ UPDTup1
_nam02_ UPDTup2
_nam03_ UPDTup3
_nam04_ AUXxxxxx
```

The suffixes up1, up2, up3, and xxxxx are _update identifier_ fields, and the fields nam00, nam01, nam02, nam03, and nam04 are _update level identifiers_.

The first record is the MACS record that defines the macro libraries (maclib1 maclib2...) to be used in the assembly in the order of search required. Up to five libraries may be specified.

Records 2, 3, and 4 are update identification records. They define the UPDTxxxx files that were created (via update control cards and source statements) to update some particular file. Record 2 defines a UPDTup1 file, and records 3 and 4 define UPDTup2 and UPDTup3 updates, respectively. None, some, or all of the updates may exist to be applied.

Record 5 defines an auxiliary file that specifies an auxiliary list of PTFs or updates that are to be applied. Record 5 defines an auxiliary file identified as 'filename AUXxxxxx', where 'filename' is the same as the filename of the input file and xxxxx is an update identifier (the update identifier for an auxiliary control file cannot be "aux"). Records in the auxiliary file have the following format for PTFs to be applied:

```
PTF  A30246CA   comments
     A21726CA
PTF  A07426CA
*    Any comment
```

The PTF field is an optional identifier, and the second field (for example, A30246CA) defines a specific PTF to be applied. The PTF has a 'filename A30246CA' identification, where 'filename' is the same as the filename of the file to be updated. The filetype of a format Axxxx6CA is used to indicate an APAR answer or PTF for APAR number xxxx. The comment field is used to describe the function of the particular PTF. The * record is ignored and is used to provide additional comments on any updates or PTFs.

The updates (PTFs included) are applied in the reverse order in which they appear. In the previous example, the updates would be applied in the following order:

```
A07426CA
A21716CA
A30246CA
UPDTup3
UPDTup2
UPDTup1
```

The PTF records can be directly included in the CNTRL file if desired, but it is usually more convenient to place them in a separate auxiliary (AUXxxxxx) file.

There can be any number of UPDTxxxx definition and auxiliary control file definition records, but only one MACS record. The complete CNTRL file can have any filename, but typically has the same name as the first specified UPDTxxxx control reccrd. In the example, the file could be named UP1 CNTRL.

The underlined fields in each record mark the level identification fields. The highest level (last) update to be applied selects the name that can be used to identify updated files. In the example, if UPDTup3 was the last update applied, then the name selected would be nam03. The value for the identification usually consists of a combination of the update identifier up1, up2, ... (up to four characters) and additional characters up to a maximum of 5 for the combined update identifier and additional characters. If no updates are applied, then the nam00 field is selected to identify the TXTnam00 produced. This name can be used to uniquely identify updated files. The text files described above, for instance, can have a filetype of TXTup3. It is desirable, on occasion, to have entries in the user CNTRL file that specify a level identification but no update. A record of the following format, for example, is allowed:

```
nam05
```

This is because the control file serves a double purpose and is used for loading text decks as well as updating input files. An identifier of TEXT as a name causes special handling in the VMFASM EXEC procedure, whether or not an update is used with it. A name of TEXT is used without level identification catenation. Thus, TEXT becomes the filetype.

SYSTEM EXEC PROCEDURES

Several system control files provide for system update and creation. Some EXEC procedures invoke others or make use of user-supplied control files to accomplish various functions such as multilevel updating, text generation, and macro library generation.

VMFASM EXEC Procedure

The VMFASM procedure performs the multilevel update function by invoking the DMSUPD module (via the CMS UPDATE command) before assembling the desired files. To update and assemble a source file, the VMFASM procedure is invoked in the following way:

```
VMFASM filename control [options]
```

where 'filename' is the name of the ASSEMBLE file to be processed and 'control' is the name of the user CNTRL file that contains the MACS (macro library), update, and any AUXxxxx control records. The VMFASM procedure invokes the DMSUPD module via the CMS UPDATE command, passing the values 'filename', 'ASSEMBLE', and 'control'.

The UPDATE command returns a level identifier and a MACLIB list from the MACS record of the control file. If the identifier is TEXT, then that becomes the filetype of the complete text deck; otherwise the filetype is TXTxxxxx (for example, TXTup3m1). The EXEC procedure then reads the MACLIB list passed by UPDATE and issues a GLOBAL command to prepare for the assembly using the specified libraries.

The ASSEMBLE program is invoked with the specified options. If no options are specified for the ASSEMBLE command, the defaults are: PRINT, NOTERM, LIST, NODECK, NORENT, SYSPARM(), and XREF(FULL). The options that can be specified for the VMFASM EXEC are: DISK, NOTERM, NOLIST, DECK, RENT, EXP, XREF, and RLD. The defaults for the VMFASM EXEC are: PRINT, TERM, LIST, NODECK, NORENT, SYSPARM(SUP), XREF(SHORT), and NORLD.

The VMFDATE program is used to construct a record for each MACLIB used and for the ASSEMBLE file. Each record is placed in the auxiliary file 'filename UPDATES'. The text deck produced by the assembler is combined with the file produced by the VMFDATE program and is named 'filename TXTxxxxx', where 'filename' is that of the ASSEMBLE file, and 'TXTxxxxx' is constructed from the update level identifier returned by the UPDATE command. All intermediate files are erased, leaving only the original ASSEMBLE and UPDTxxxx files, and the newly created text file.

GENERATE EXEC Procedure

The GENERATE procedure is generally used during system generation. It can build a CP, CMS, or RSCS nucleus and punch or create self-loading card decks for the four standalone service programs (DMKDIR, DMKDDR, DMKFMT, and IBCDASDI). GENERATE can also build a new VM/370 directory, a new real I/O deck (DMKRIO), a new buffer load (DMKFCB), a new system name table (DMKSNT), or a new system deck (DMKSYS). GENERATE can also load the IPCS modules from tape onto the IPCS A-disk. The GENERATE procedure uses the VMFASM EXEC procedure to reassemble DMKRIO, DMKFCB, DMKSNT, and DMKSYS. It also uses the VMFLOAD program to build the CP, CMS, or RSCS nucleus.

VMFLOAD SERVICE PROGRAM

The VMFLOAD program uses two user-supplied procedures, a loadlist EXEC and a 'control' file identical in format to the CNTRL file used by VMFASM and UPDATE, to produce a punched deck comprised of several text files. The VMFLOAD program is invoked as a CMS command in the following way:

```
VMFLOAD loadlist control
```

The loadlist is a user-supplied EXEC file consisting of several records of the following format:

```
&CONTROL OFF
&1 &2 &3 filename [filetype]
&2 &2 &3 filename [filetype]
      .
      .
      .
```

The 'filename' specifies the name of a text file to be punched. The text files are punched in the order specified. If a filetype is specified, a search is made for that specific file, and if it is found it is punched without a header card, and the search then bypasses the control file.

If the filetype is not given, the specified control file is used to search for the highest level text file available, and it is punched.

The VMFLOAD program displays a confirmation or error message upon completion. Before invoking the loadlist procedure, a SPOOL PCH CONT command line is executed to assure that the punched files appear as one deck. The command lines SPOOL PCH NOCONT and CLOSE PCH are executed upon completion.

The control field is used only if the filetype is not specified. The control field specifies a user-supplied control file with a filename of 'control' and a filetype of CNTRL. This control file is of the same type and format as the one used to perform multilevel updates. Indeed, most often the file used to produce the updated and assembled text decks is the one used to load the text decks.

VMFLOAD uses the control file to search for the desired text deck in the order in which the identifiers are specified in the file. The first file located is punched, and all lower files are ignored. If the end is reached without finding a text file, VMFLOAD displays the message 'filename TEXT' NOT FOUND, and continues processing with the next entry in the loadlist EXEC. It is quite possible to have a completed load deck comprised of different levels of text decks.

DMKLD00E SERVICE PROGRAM

The loader (DMKLD00E) is a service program that is used to generate a CP, CMS, or RSCS nucleus. The loader loads the text decks supplied with it, resolves CCW addresses, and resolves address constants. The same loader is used whether a virtual=real or standard CP system is generated.

The loader is distributed with the following default I/O addresses:

```
Console=009
Printer=00E
```

These addresses can be overridden by a control card that must be placed between the loader END card and the first card of the text decks. The format of the control card is:

Column	Contents
1	0-2 punch
2-4	DEV
5	blank
6-13	PRNT=xxx (xxx is the printer address)
14	blank
15-22	TYPW=xxx (xxx is the console address)

The format of the other control cards can be found in the discussion of the LOAD command in the _VM/370 CMS Command and Macro Reference_.

The loader is self-relocating, that is, it is initially loaded at address 8000 (decimal); it then relocates itself to the top of storage. (For example, if the size of the loader is 10K, and the storage size of the system is 256K, the loader will occupy the area of storage between 246K and 256K.) After relocating itself, the loader clears the storage it was originally loaded in. As the loader needs free storage to perform its operations, it extends downward through storage.

The text decks being loaded must not try to overlay either the loader or any address between zero and 100 (hexadecimal). The text decks are loaded into storage in a positive direction (that is, upward through storage). If the text decks are going to overlay the loader's free storage, the operation is terminated.

THE VMFMAC MACRO LIBRARY UPDATE PROCEDURE

The VMFMAC procedure applies updates to copy or macro files and builds a new macro library. The VMFMAC EXEC procedure is invoked with the following command line:

```
VMFMAC maclibname cntrlname
```

where:

maclibname is the filename of the file that contains a list of the macro and copy files that are to be included, or updated and included, in the new macro library. This list file must have a filetype of EXEC and each entry in the maclibname EXEC file has the following format:

```
&1 &2 filename1
&1 &2 filename2
      .
      .
      .
```

cntrlname is the filename of the control file used to apply the updates. The control file (filetype CNTRL) may contain the actual update or only the names of other files that contain the updates.

The UPDATE command is issued for each macro or copy file. If the update procedure is successful, the member is added to the NEWMAC MACLIB. After all macro and copy files have been processed, any existing libname MACLIB file is erased and the NEWMAC MACLIB is renamed to libname MACLIB.

Method of Operation

This section describes the following procedures fcr generating and updating VM/370:

- Update procedure
- Nucleus loading facility
- The MACLIB generation facility

Figure 7-1 shows the relationship of the diagrams.

Diagram 7-1 shows the major functions of the VMFASM procedure.

Diagram 7-2 shows the initialization of the VMFASM procedure.

Diagram 7-3 describes the assembling portion of the VMFASM procedure.

Diagram 7-4 describes the VMFDATE program.

Diagram 7-5 describes the major functions of the DMSUPD (update) program.

Diagram 7-6 describes the operand and option checking for the Update program.

Diagram 7-7 describes the multiple level update procedure.

Diagram 7-8 describes the processing of control records for the Update program.

Diagram 7-9 describes the single level update procedure.

Diagram 7-10 shows how inserting is done.

Diagram 7-11 describes the exit procedure for the Update program.

Diagram 7-12 describes the module load program.

Diagram 7-13 describes the procedure that builds the MACLIB.

Diagram 7-14 describes the major functions of the GENERATE procedure.

Diagram 7-15 describes the CP portion of the GENERATE procedure.

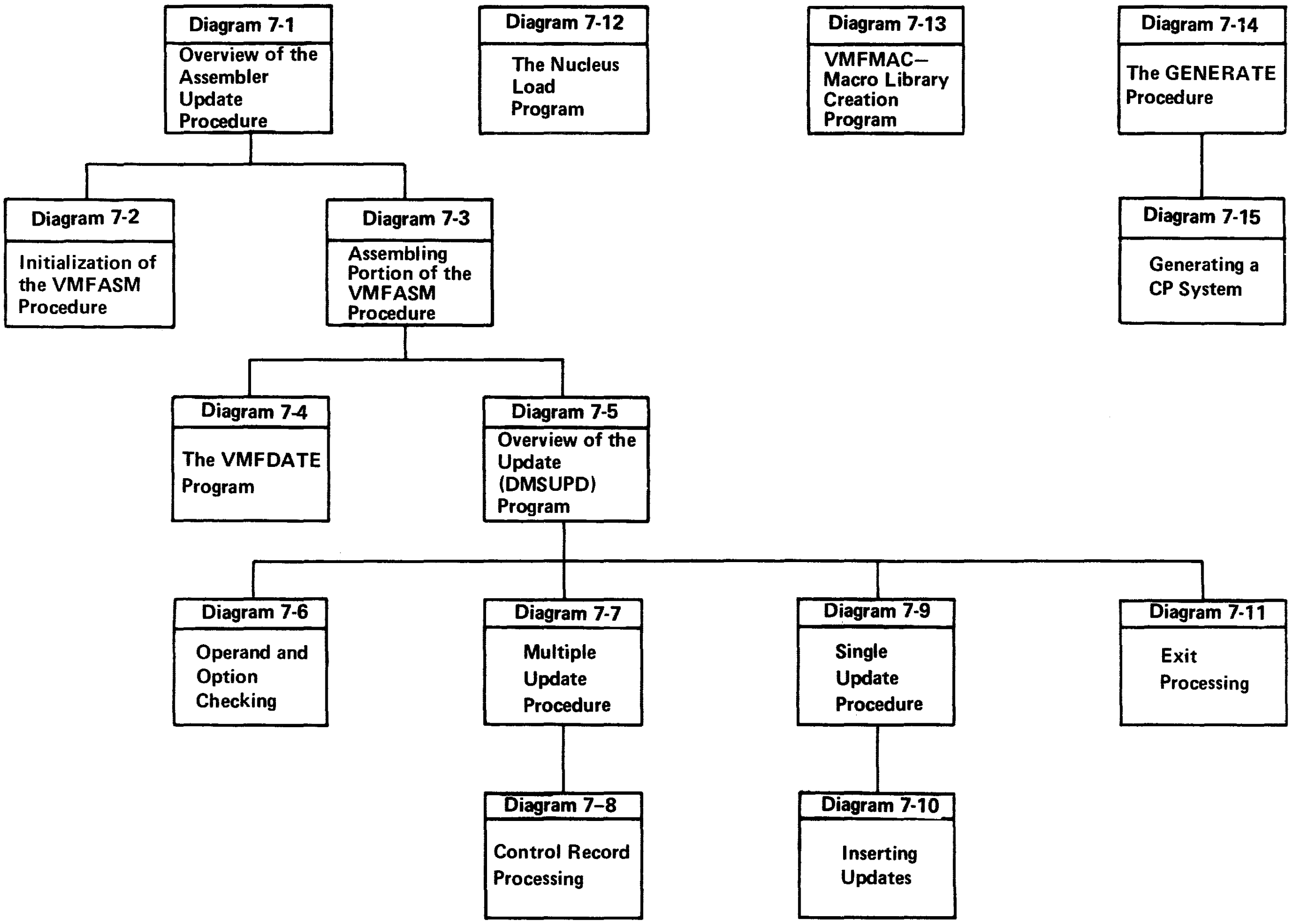
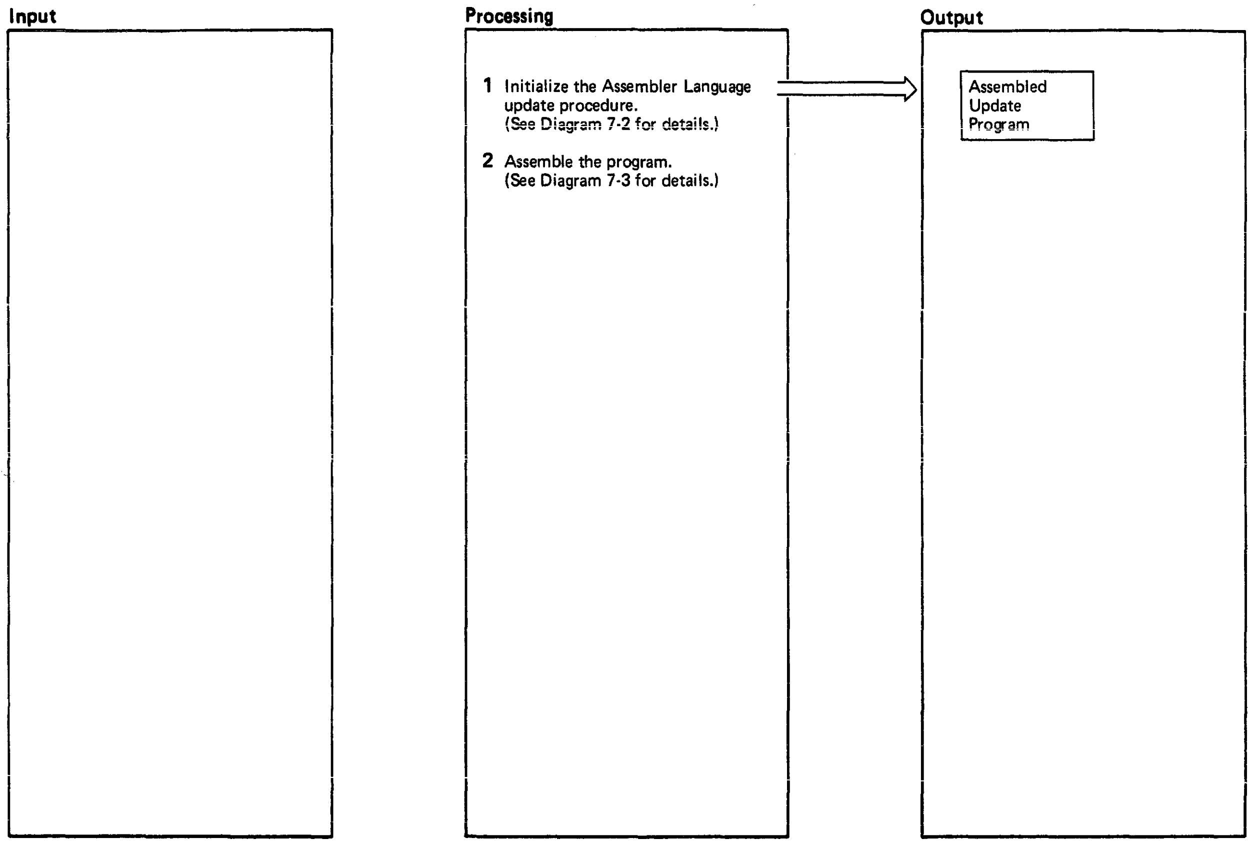


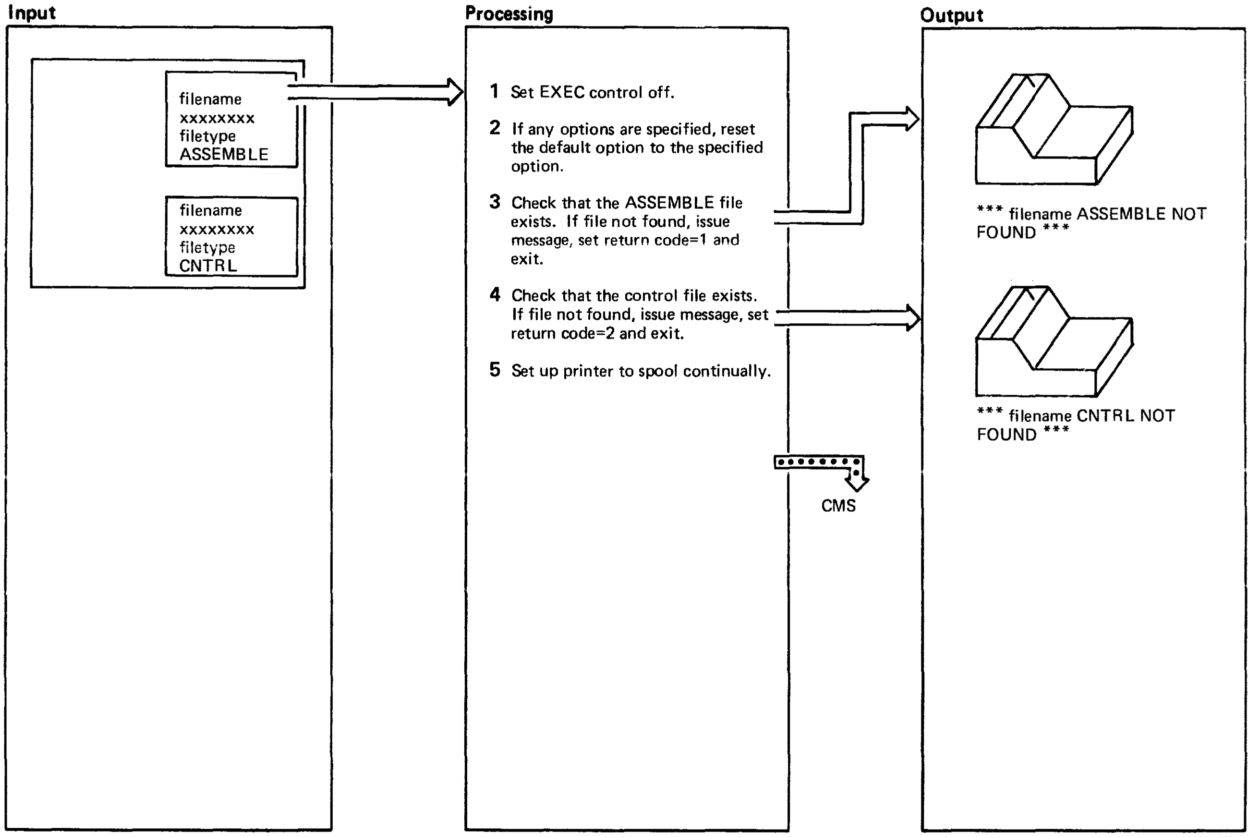
Figure 7-1. Key to the Procedures for Generating and Updating VM/370 Method of Operation Diagrams



1 Initialize the Assembler Language update procedure. (See Diagram 7-2 for details.)

2 Assemble the program. (See Diagram 7-3 for details.)

Diagram 7-1. Overview of the Assembler Update Procedure



1 Set EXEC control off.

2 If any options are specified, reset the default option to the specified option.

3 Check that the ASSEMBLE file exists. If file not found, issue message, set return code=1 and exit.

4 Check that the control file exists. If file not found, issue message, set return code=2 and exit.

5 Set up printer to spool continually.

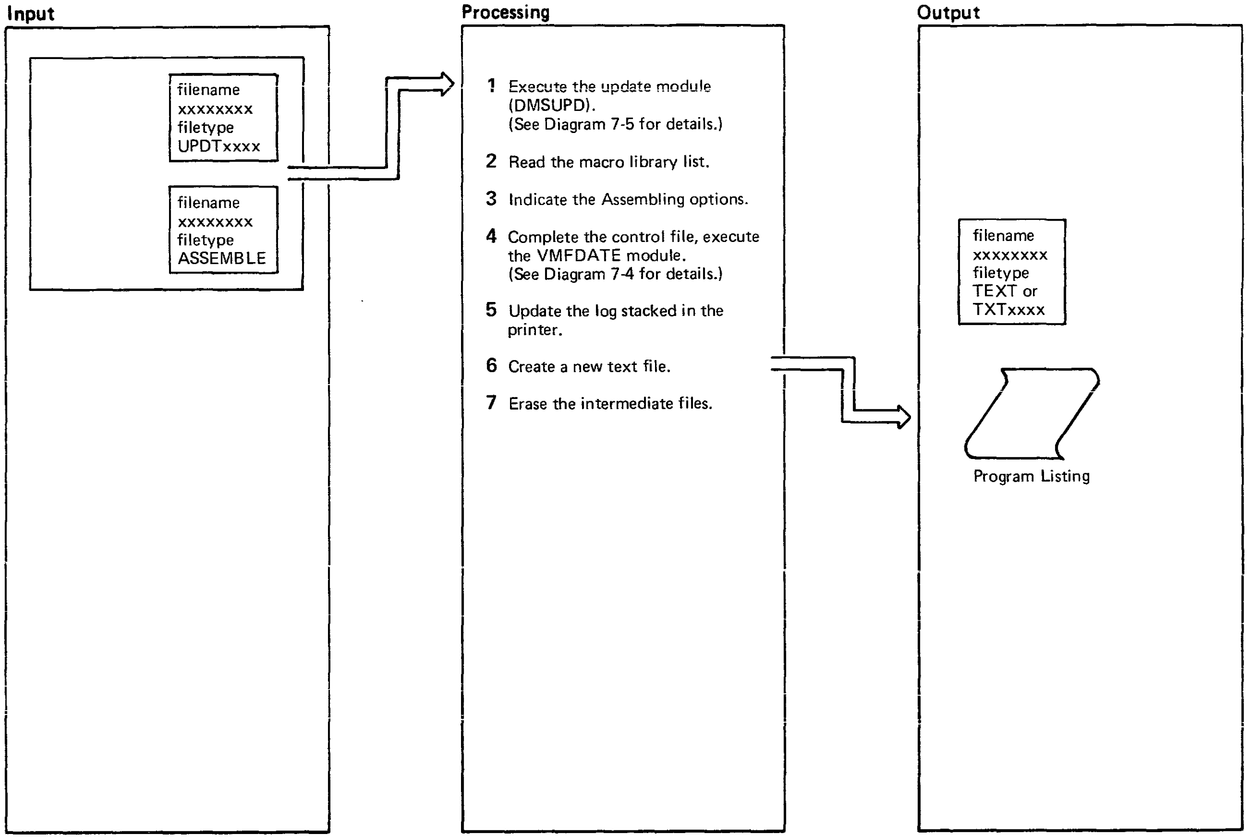
*** filename ASSEMBLE NOT FOUND ***

*** filename CNTRL NOT FOUND ***

Notes	Module	Label	Ref
1 The CMS commands executed and the return codes that result will not be displayed on the virtual machine console.	VMFASM		
2 The default options are: PRINT, TERM, LIST, NODECK, NORENT, SYSPARM(SUP), XREF(SHORT), and NORLD. The options specified for the VMFASM EXEC are: DISK, NOTERM, NOLIST, DECK, RENT, EXP, XREF, and RLD.	VMFASM		
3 The CMS STATE command is executed. A nonzero return code indicates that the ASSEMBLE file was not found.	VMFASM	-STSYS	
4 The CMS STATE command is executed. A nonzero return code indicates that the CNTRL file was not found.	VMFASM	-STCTL	
5 The CP SPOOL command is executed.	VMFASM	-FUPD	

Notes	Module	Label	Ref

Diagram 7-2. Initialization of the VMFASM Procedure



1 Execute the update module (DMSUPD). (See Diagram 7-5 for details.)

2 Read the macro library list.

3 Indicate the Assembling options.

4 Complete the control file, execute the VMFDATE module. (See Diagram 7-4 for details.)

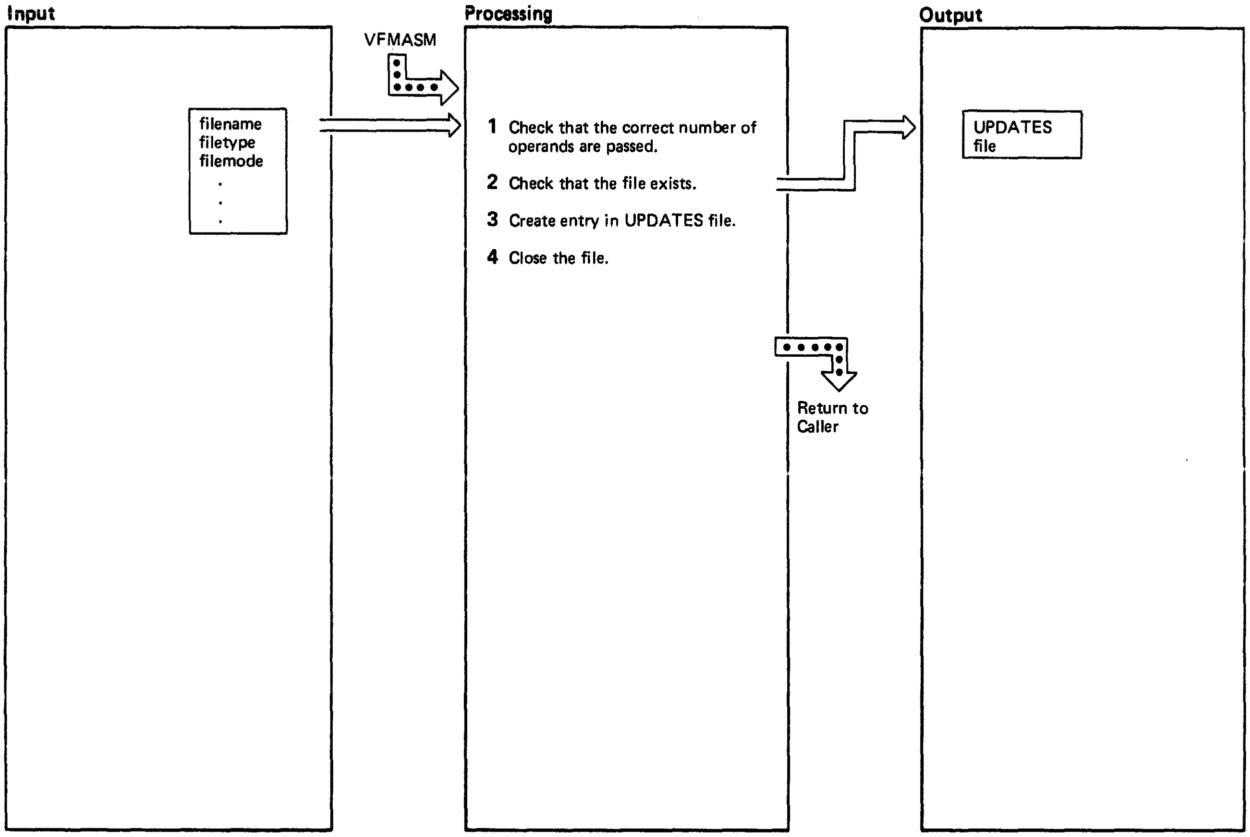
5 Update the log stacked in the printer.

6 Create a new text file.

7 Erase the intermediate files.

Notes	Module	Label	Ref
1 The DMSUPD module is executed. The name of the ASSEMBLE and CNTRL files and a filetype of ASSEMBLE are passed to the DMSUPD module. The DMSUPD module returns a level identifier and a MACLIB (macro library) list. A return code between 20 and 36 causes the VMFASM EXEC procedure to display the message *** ERROR UPDATING filename and return control to the CMS command environment. If the level identifier is TEXT, TEXT becomes the filetype of the completed text deck. If the level identifier (xxxxx) is not TEXT, the filetype becomes TXTxxxxx. If the return code is 40 (no updates), the filename is the same as the filename of the original ASSEMBLE file. Otherwise, the filename is set to the updated filename.	VMFASM	-FUPD	
2 The MACLIB list is read. The VMFDATE module is executed once for each MACLIB. The CMS GLOBAL command is issued to identify the macro libraries that will be used during the assembly.	VMFASM		
3 If any options were specified on the VMFASM command, the message ASMBLING filename (options . . .) is displayed indicating the specified options.	VMFASM		
If no options were specified on the VMFASM command, the default options are assumed and the message ASMBLING filename is displayed.		-ASMP	
4 The VMFDATE module is executed once more to complete the UPDATES file.	VMFASM	-DTF	
5 The UPDATES file is printed on the virtual printer and then erased.	VMFASM	-DTF	
6 The updated file is assembled. If ASSEMBLE returns a nonzero code, the message *** ERROR ASMBLING filename *** is displayed. The STATE command is issued to see if a text deck actually exists. If the text deck does not exist, the message *** NO TEXT FOR filename *** is displayed, the VMFASM EXEC procedure terminates, and control returns to the CMS command environment.	VMFASM	-DTF	
7 The new text file, original ASSEMBLE file, and any UPDTxxxx files are saved. The message filename $\left\{\begin{matrix}\text{TEXT}\\\text{TXTxxxx}\end{matrix}\right\}$ CREATED is displayed. All intermediate files are erased. The printer is closed and control returns to the CMS command environment.	VMFASM	-COMB -EXIT	

Diagram 7-3. Assembling Portion of the VMFASM Procedure



1 Check that the correct number of operands are passed.

2 Check that the file exists.

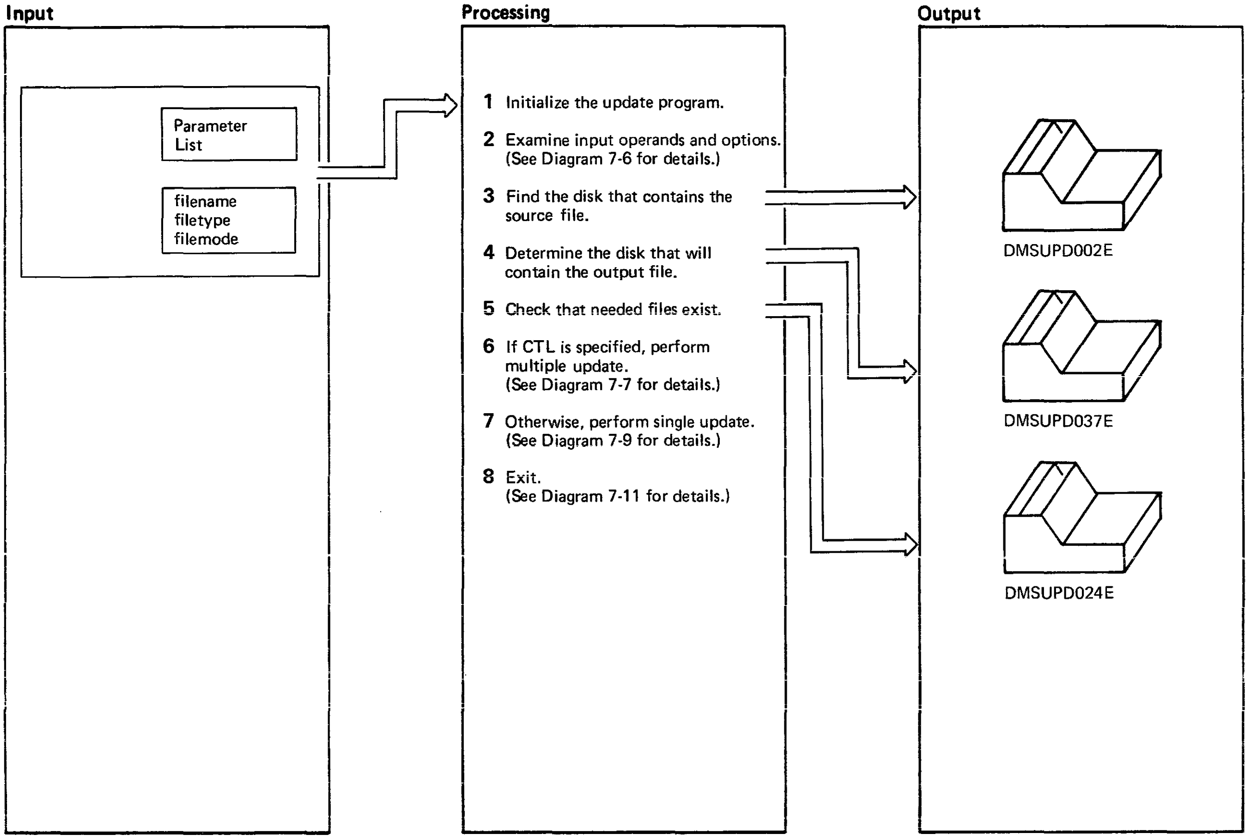
3 Create entry in UPDATES file.

4 Close the file.

Notes	Module	Label	Ref
1 Six operands should be passed to the VMFDATE module. The first three operands are the filename, filetype, and filemode of the input file. The next three operands are the filename, filetype, and filemode of the output file.	VMFDATE	VMFDATE	
2 If the input file does not exist, control returns to the calling routine.	VMFDATE	TEST	
3 Each time the VMFDATE module is called, it creates an entry in the VMCNTRL file indicating that an update was applied. The format of each entry is: * filename filetype filemode volid date time The disk label is picked up from the ADT (Active Disk Table).	VMFDATE		
4 The UPDATES file is closed and control returns to the calling routine.	VMFDATE		

Notes	Module	Label	Ref

Diagram 7-4. The VMFDATE Program



1 Initialize the update program.

2 Examine input operands and options. (See Diagram 7-6 for details.)

3 Find the disk that contains the source file.

4 Determine the disk that will contain the output file.

5 Check that needed files exist.

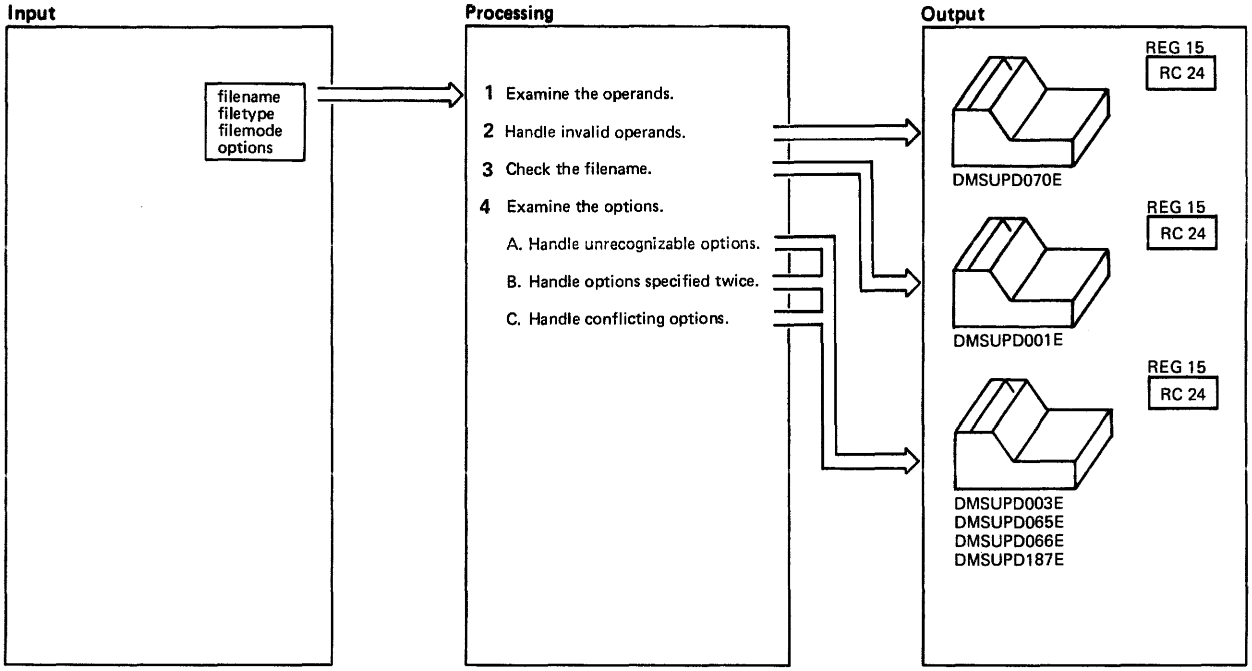
6 If CTL is specified, perform multiple update. (See Diagram 7-7 for details.)

7 Otherwise, perform single update. (See Diagram 7-9 for details.)

8 Exit. (See Diagram 7-11 for details.)

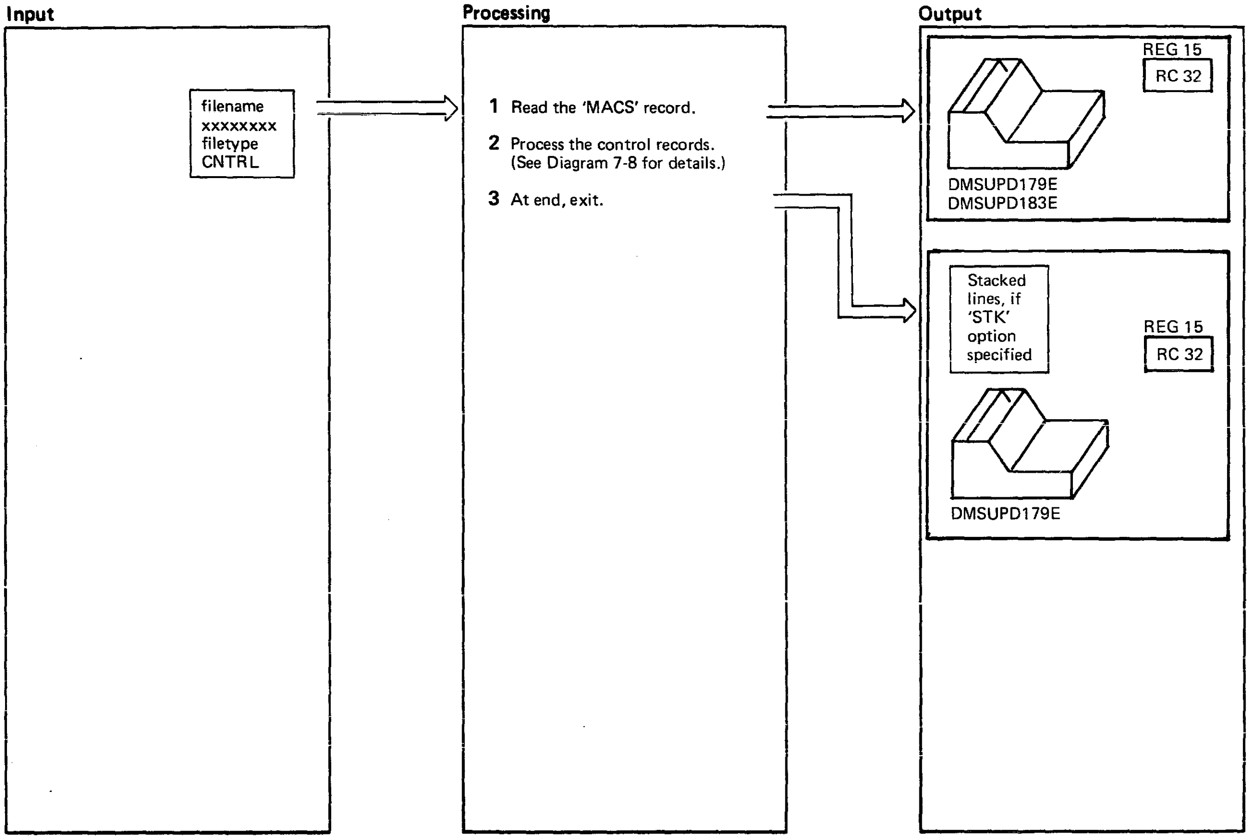
Notes	Module	Label	Ref
1 Registers 12, 11, and 9 are set up as base registers. All indicators are set off.	DMSUPD	DMSUPD	
2 The filename operand is required.	DMSUPD	DMSUPD	
3 DMSUPD checks that the source input file exists. If not, the message DMSUPD002E FILE 'fn ft fm' NOT FOUND is displayed and control returns to the CMS command environment with a return code of 28 in register 15.	DMSUPD	PROCESS NOFILE	
4 The DMSUPD module searches for a suitable disk to hold the output files. First, an attempt is made to place the files on the same disk that contains the original input. If the input disk is read-only, but is an extension of a read/write disk, an attempt is made to place the files on that disk. Lastly, an attempt is made to place the files on the A-disk. If all these attempts fail, the message DMSUPD037E DISK 'A' IS READ/ONLY is displayed and control returns to the CMS command environment with a return code of 36 in register 15.	DMSUPD	PROCESS	
5 DMSUPD issues the STATE command to see if the UPDATE CMSUT1 file already exists: it should not exist. If the CMSUT1 file exists, the message DMSUPD024E FILE 'UPDATE CMSUT1 fm' ALREADY EXISTS is displayed and control returns to the CMS command environment with a return code of 24 in register 15.	DMSUPD	PROCESS	
If the DISK option was specified, an old copy of 'filename UPDLOG' is erased (if one exists).			
If the control file option (CTL) is specified DMSUPD checks that the control file exists and continues processing at the CTLMULT (multiple update) routine.		NOERASE	
If the control file option is not specified, DMSUPD checks that the single update file exists and continues processing at the single update (SINGUPD) routine.		LOCTUPD	
6 See Diagram 7-7.	DMSUPD	CTLMULT	
7 See Diagram 7-9.	DMSUPD	SINGUPD	
8 See Diagram 7-11.	DMSUPD	RETR001	

Diagram 7-5. Overview of the Update (DMSUPD) Program



Notes	Module	Label	Ref
1 DMSUPD uses the filename operand to set up the disk parameter lists for input, update log, and auxiliary files. All the operands (except the required filename) and all the options are read by branching and linking to the OPTSCAN routine. The first three operands are the filename, filetype, and filemode of the file to be updated. The next three operands are the filename, filetype, and filemode that describe the update or control file to be applied.	DMSUPD	DMSUPD	
2 If more than six operands are specified before the left parenthesis, the message DMSUPD070E INVALID PARAMETER 'param' is displayed and control returns to the CMS command environment with a return code of 24 in register 15.		EXCESIV	
3 Only the first operand must be specified. If no operands are found, the message DMSUPD001E NO FILENAME SPECIFIED is displayed and control returns to the CMS command environment with a return code of 24 in register 15.	DMSUPD	NOFNAME	
4 The options assumed, if not otherwise specified are: SEQ8, NOINC, NOREP, NOCTL, NOSTK, TERM, and DISK. When the last option is processed, control returns to the PROCESS routine.	DMSUPD		
A. If an unrecognizable option is specified, the message DMSUPD003E INVALID OPTION 'option' is displayed and control returns to the CMS command environment with a return code of 24 in register 15.		INVOPTN	
B. If an option is specified twice, the message DMSUPD065E 'option' OPTION SPECIFIED TWICE is displayed and control returns to the CMS command environment with a return code of 24 in register 15.		OPTDUP	
C. If two conflicting options are specified, the message DMSUPD066E 'option' AND 'option' ARE CONFLICTING OPTIONS is displayed and control returns to the CMS command environment with a return code of 24 in register 15. The conflicting pairs of options are: SEQ8, and NOSEQ8, INC and NOINC, REP and NOREP, STK and NOSTK, TERM and NOTERM, CTL and NOCTL, CTL and NOINC, and DISK and PRINT.		OPTCONF	
If the STK option is specified without the CTL option, the message DMSUPD187E OPTION 'STK' INVALID WITHOUT 'CTL' is displayed, and control returns to the CMS command environment with a return code of 24 in register 15.		ERSC	

Diagram 7-6. Operand and Option Checking



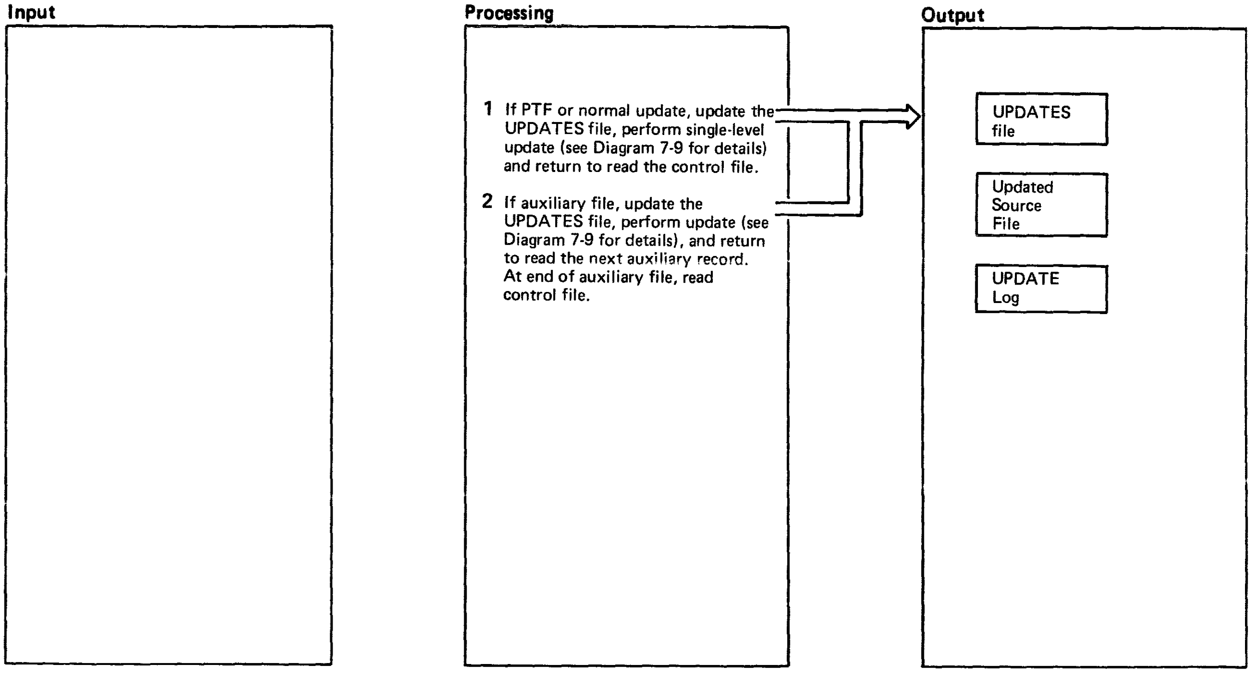
1 Read the 'MACS' record.

2 Process the control records. (See Diagram 7-8 for details.)

3 At end, exit.

Notes	Module	Label	Ref
1 The macro library (MACS) record is read from the beginning of the control file and saved. If the MACS card is not found, or is not the first noncomment card in the control file, the message DMSUPD179E MISSING OR DUPLICATE 'MACS' CARD IN CONTROL FILE 'fn ft fm' is displayed and control returns to the CMS command environment with a return code of 32 in register 15.	DMSUPD	CTMULT ERMACS	
If the MACS control card is invalid, the message DMSUPD183E INVALID CONTROL FILE CONTROL CARD is displayed and control returns to the CMS command environment with a return code of 32 in register 15.		BATCTLC	
2 See Diagram 7-8.	DMSUPD	CTLGETM	
3 If a 'MACS' record is read, the file is completely processed. The control file is closed.	DMSUPD	CTLDONE	
If this MACS card does not have an item number identical to that of the MACS control card originally read, the control file contains duplicate MACS control cards. The message DMSUPD179E MISSING OR DUPLICATE 'MACS' CARD IN CONTROL FILE 'fn ft fm' is displayed and control returns to the CMS command environment with a return code of 32 in register 15.		ERMACS	
If STK is specified, the updated level ID is stacked in the terminal input stack.			

Diagram 7-7. Multiple Update Procedure



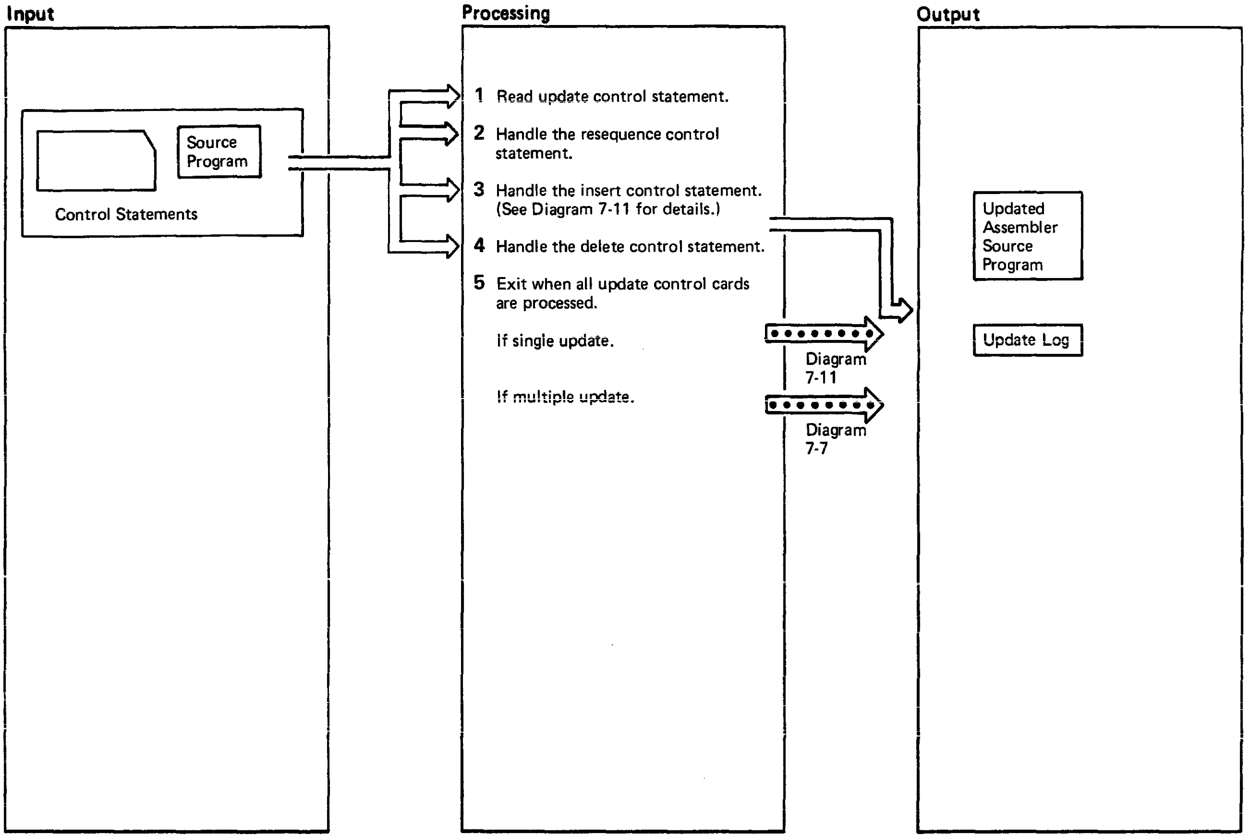
1 If PTF or normal update, update the UPDATES file, perform single-level update (see Diagram 7-9 for details) and return to read the control file.

2 If auxiliary file, update the UPDATES file, perform update (see Diagram 7-9 for details), and return to read the next auxiliary record. At end of auxiliary file, read control file.

Notes	Module	Label	Ref
1 The control file is read from the bottom up. If the control record is valid, the message	DMSUPD	CTLGETM CTLREAD	
DMSUPD183E INVALID CONTROL FILE CONTROL CARD is displayed, and control returns to the CMS command environment with a return code of 32 in register 15.		BADCTLC	
If the PTF or update file is not found, control returns to the read routine (CTLREAD). If the file is found and the update is not being performed in storage, the message		CTLIPTF CTLOCUP CTLUMSG	
DMSUPD178I UPDATING 'fn ft fm' WITH 'fn ft fm' is displayed and an entry is made in the UPDATES file. If the update is being performed in storage, free storage is acquired to contain the input file. The message		CTLUMSS	
DMSUPD300E INSUFFICIENT STORAGE TO BEGIN UPDATE is displayed if the input file is too large for the acquired storage.		SMALLCOR	
If the STOR option was not specified explicitly, the message DMSUPD304E UPDATE PROCESSING WILL BE DONE USING DISK is also displayed. If the STOR option was specified, control returns to CMS with a return code of 40 in register 15. If processing continues, the input file is read into the acquired storage, the message		IMPLICIT	
DMSUPD178I UPDATING 'fn ft fm' WITH 'fn ft fm' is displayed, and an entry is made in the UPDATES file.		CTLUMSS	

Notes	Module	Label	Ref
Then a branch to the SINGUPD routine transfers control to the single update routine. After the update is performed, control returns to CTLCONT.			
2 DMSUPD checks that the auxiliary file exists. If not, control returns to the read routine (CTLREAD). If the auxiliary file is found, it is read from the bottom up.	DMSUPD	AUXFIND	
If the PTF file within the auxiliary file is not found, the message DMSUPD180W MISSING PTF FILE 'fn ft fm' is issued. The RETCODE value is set to 12 if it has not been set higher previously. Processing continues with the next record from the auxiliary file (AUXREAD).		NOFILEW	
When a valid record is read from the auxiliary file, the message		CTLUMSG	
DMSUPD178I UPDATING 'fn ft fm' WITH 'fn ft fm' is displayed and an entry is made in the UPDATES file. Then the SINGUPD routine applies the update. After the update is performed, control returns to CTLCONT which returns control to		CTLUMSS	
AUXREAD. This loop continues until the entire auxiliary file is processed.		AUXREAD	
At the end of the auxiliary file, the file is closed and control returns to the control file read routine (CTLREAD).		AUXFINT	
If an invalid card is found in the auxiliary file, the message DMSUPD183E INVALID AUX FILE CONTROL CARD is displayed and control returns to the CMS command environment with a return code of 32 in register 15.		BADAUXC	

Diagram 7-8. Control Record Processing



1 Read update control statement.

2 Handle the resequence control statement.

3 Handle the insert control statement. (See Diagram 7-11 for details.)

4 Handle the delete control statement.

5 Exit when all update control cards are processed.

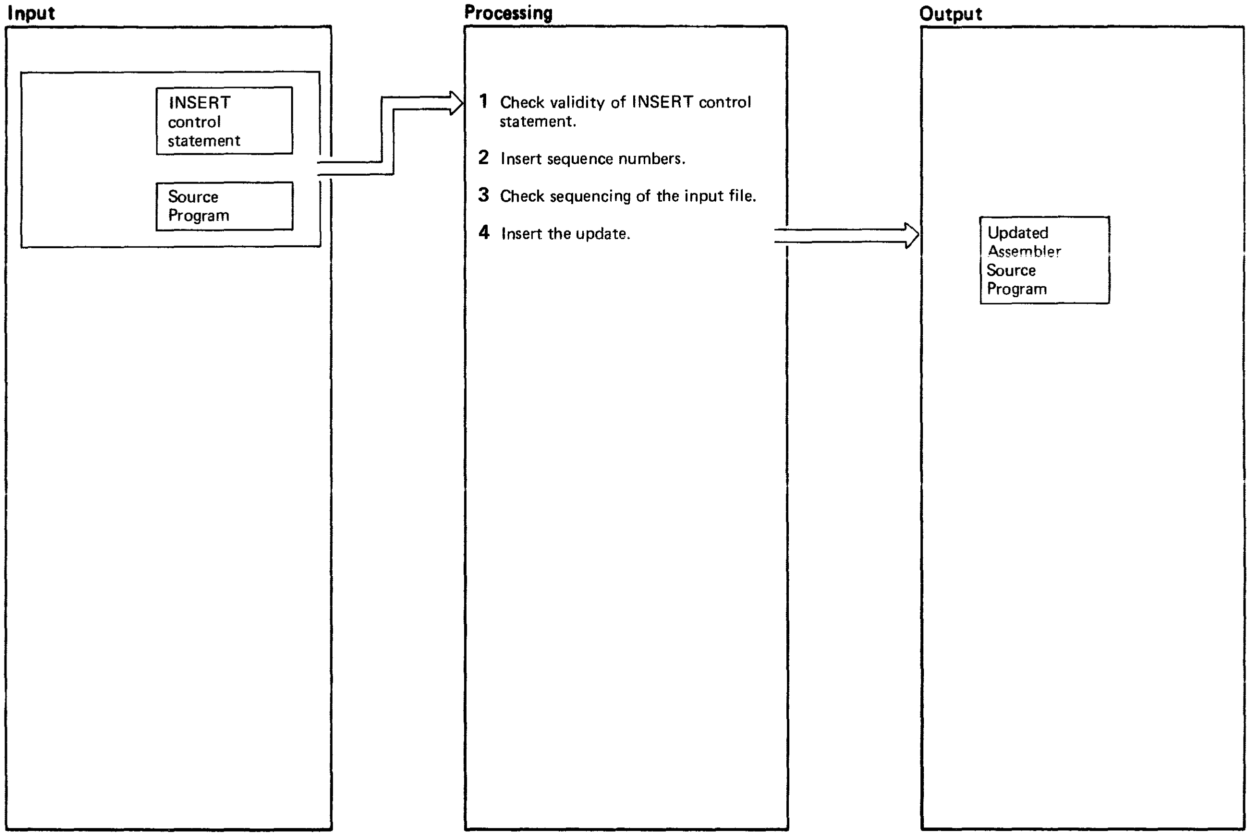
If single update.

If multiple update.

Notes	Module	Label	Ref
1 An update card is read and checked. If an invalid control card is read, the message DMSUPD207W INVALID UPDATE FILE CONTROL CARD is issued. The value of RETCODE is set to 12, if it was not previously set higher. Processing continues ignoring the invalid card.	DMSUPD	SINGUPD	
2 DMSUPD checks the resequence card. If the resequence card is not the first card in the update file, the message DMSUPD184W './S' NOT FIRST CARD IN UPDATE FILE – IGNORED is issued. The value in RETCODE is set to 12 if it has not been set higher previously. The './S' card is ignored and processing continues.	DMSUPD	FCTRSEQ RSEQERR	
If an invalid character is specified in one of the sequence fields, the message DMSUPD185W INVALID CHAR IN SEQUENCE FIELD 'xxxxxxxx' is issued. The value of RETCODE is set to 12 if it was not set higher previously. The './S' card is ignored and processing continues.		INVCHAR	
If the specified sequence increment is zero, the message DMSUPD182W SEQUENCE INCREMENT IS ZERO is issued. The value of RETCODE is set to 8 if it has not been set higher previously. Processing continues and the file is resequenced with a sequence increment of zero.		ZERSEQ	

Notes	Module	Label	Ref
If no errors are found, the sequencing is set to 5 or 8 characters depending on the options specified (SEQ8 or NOSEQ8). The UPDFLAG is set for resequencing and the next update control card is read (UPDREAD).		RSEQDEF RSEQFIN	
3 See Diagram 7-11.	DMSUPD	FCTINST	
4 The update control card is checked. The indicated cards are removed. The control statement and the message DELETING . . . are sent to the UPDLOG file. If the delete is being performed in storage, the records in storage are rechained, eliminating the deleted records.	DMSUPD	FCTDELT DELTINE XDELE	
5 When all the update control cards are processed, the UPDREAD (read) routine takes its error exit (UPDFERR). The UPDFERR routine branches to the INPUTRD routine on an end-of-file condition to flush (write out) the rest of the input source file if the update was not performed in storage. If the update was performed in storage, and resequencing is requested, a logical replace is done on each line in the file. The error exit (INPFERR) is taken from the INPUTRD routine. The INPFERR routine closes the updated file and the input file. If processing a control file (multiple update), control returns to CTLCONT. Otherwise, the single-level update is complete and control is returned to CMS (RRETURN exit routine).	DMSUPD	UPDREAD XDELE	

Diagram 7-9. Single Update Procedure



1 Check validity of INSERT control statement.

2 Insert sequence numbers.

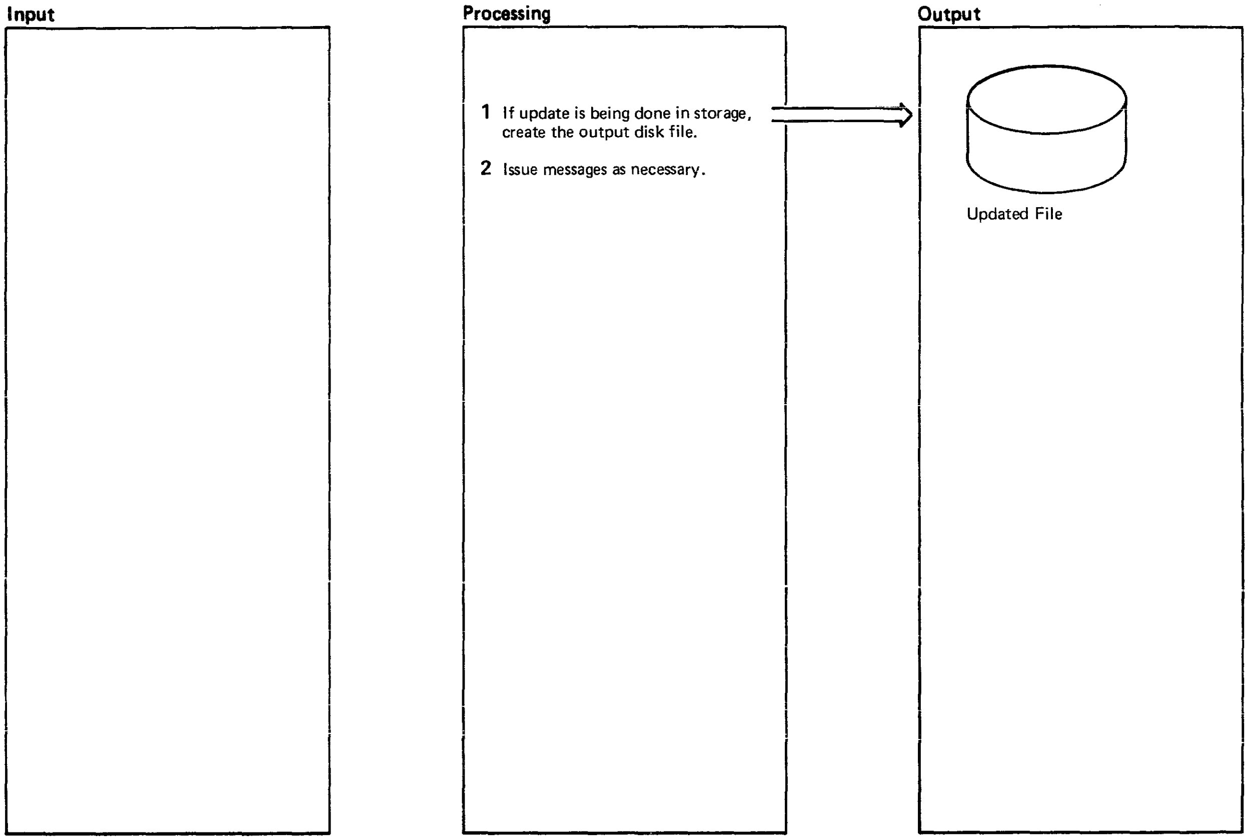
3 Check sequencing of the input file.

4 Insert the update.

Notes	Module	Label	Ref
1 The INSERT card is checked. If invalid, the message DMSUPD207W INVALID UPDATE FILE CONTROL CARD is issued. The value of RETCODE is set to 12 if was not set higher previously. The invalid card is ignored and processing continues.	DMSUPD	FCTINST INVUPCD	
2 If requested, the sequence numbers are put in the inserts. Otherwise, the sequence number field contains ********	DMSUPD	FCTREPL	
If a specified sequence number is not found, the message DMSUPD186W SEQUENCE NUMBER 'xxx' NOT FOUND is issued. The value of RETCODE is set to 12 if it has not been set higher previously. The invalid card is ignored and processing continues.		UPDSERR	
3 If the input file sequence numbers are out of order, the message DMSUPD210W INPUT FILE SEQUENCE ERROR 'xxx' TO 'xxx' is issued. The value of RETCODE is set to 4 if it was not set higher previously. Processing continues.	DMSUPD	INSEQW	
4 DMSUPD inserts the cards. The control statement and the INSERTING . . .	DMSUPD	INSLOOP	

Notes	Module	Label	Ref
message are sent to the 'UPDLOG' file.			
If the sequence errors are introduced in the output file, the message DMSUPD174W SEQUENCE ERROR INTRODUCED IN OUTPUT FILE 'xxx' TO 'xxx' is issued. The value of RETCODE is set to 8 if it was not set higher previously. Processing continues.			
If sequence overflow occurs while cards are being inserted, the message DMSUPD176W SEQUENCE OVERFLOW FOLLOWING SEQUENCE NUMBER 'xxx' is issued. The value of RETCODE is set to 8 if it was not previously set higher. Processing continues.		WOVF	
When the appropriate cards are successfully inserted in the file, control returns to the read routine to read the next control card.			

Diagram 7-10. Inserting Updates



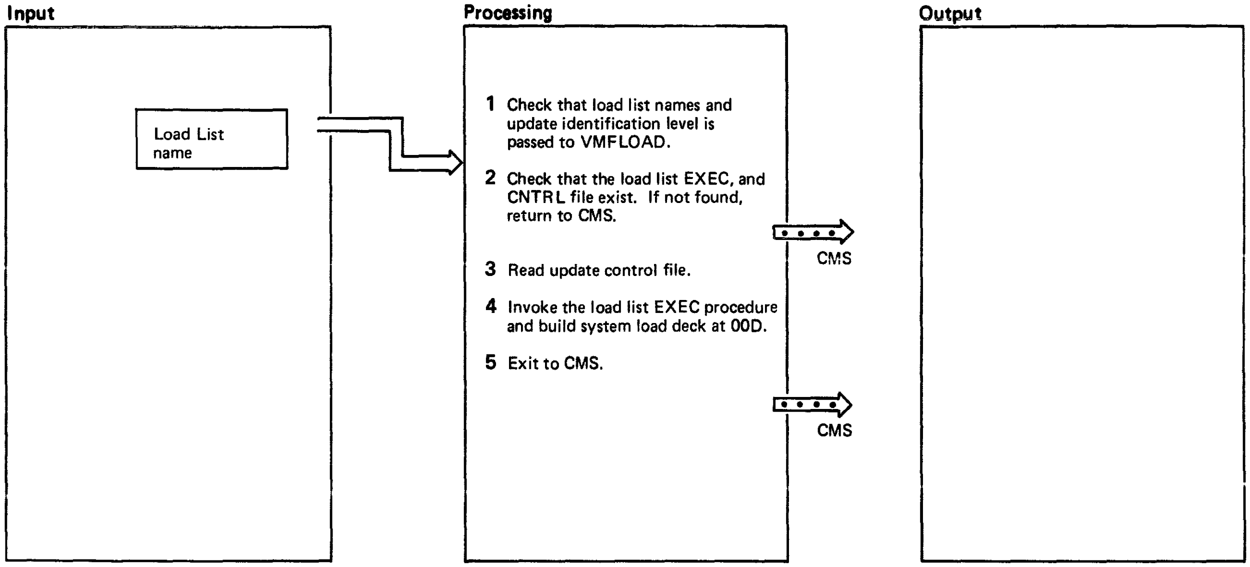
1 If update is being done in storage, create the output disk file.

2 Issue messages as necessary.

Notes	Module	Label	Ref
1 If the update is being performed in storage, the updated file in storage is read line by line and a disk file is created with the filename and filetype UPDATE CMSUT1. The filemode specifies the disk where the final output file resides. The disk file is then	DMSUPD	RETR001	
closed. The UPDATE CMSUT1 file is then renamed $fname after the old $fname is erased.		RETRD	
2 If RETCODE is not equal to zero, warning messages were issued during the update.	DMSUPD		
If warning messages are issued and the NOTERM option is specified, while the REP option is not, the message DMSUPD177I WARNING MESSAGES ISSUED (SEVERITY=nn) is displayed (nn is the value in RETCODE). If warning messages are issued and the REP option is specified, whether or not the NOTERM option is specified, the message DMSUPD177I WARNING MESSAGES ISSUED (SEVERITY = nn) 'REP' OPTION IGNORED is displayed (nn is the value of RETCODE). In either case, control returns to the CMS command environ-		WRETURN	

Notes	Module	Label	Ref
ment with the value of RETCODE in register 15. If no warning messages are issued and the REP option is specified, the '$fname' file is renamed to 'fname', after the old file is erased. If the CTL option is specified and no update files are found, the message			
DMSUPD181E NO UPDATE FILES WERE FOUND is displayed and control returns to the CMS command environment with a return code of 40 in register 15. If no warning messages are issued, and no errors detected, control returns to the CMS command environment with a return code of 0 in register 15.		NOUPDATS	

Diagram 7-11. Exit Processing



1 Check that load list names and update identification level is passed to VMFLOAD.

2 Check that the load list EXEC, and CNTRL file exist. If not found, return to CMS.

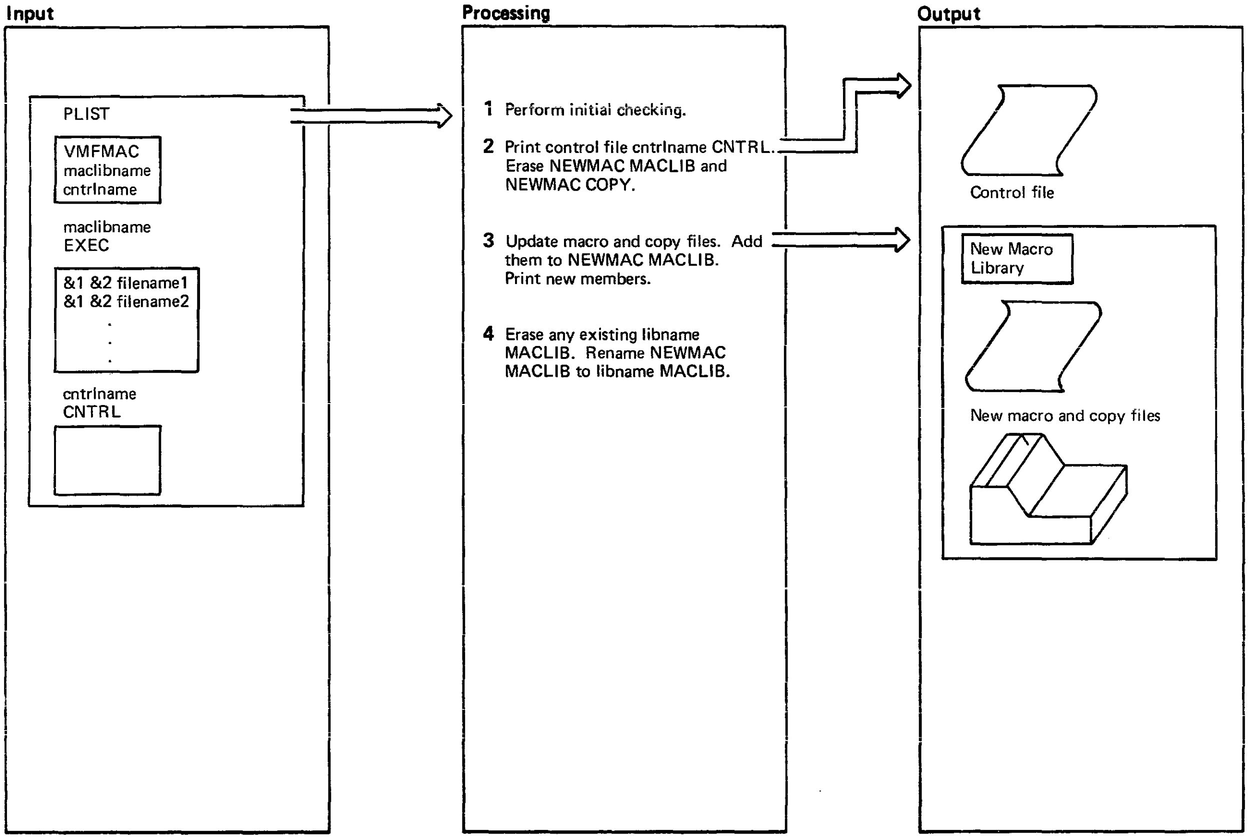
3 Read update control file.

4 Invoke the load list EXEC procedure and build system load deck at 00D.

5 Exit to CMS.

Notes	Module	Label	Ref
1 The load list name is moved into the filename portion of a STATE command line for an EXEC file and a CNTRL file.	VMFLOAD	VMFLOAD	
The update identification level is moved into the filename portion of a RDBUF command line for an EXEC and CNTRL file.			
2 Issue the STATE command via an SVC 202 to make sure that the load list EXEC and CNTRL files exist.	VMFLOAD		
If the load list EXEC file is not found, the message NO LOAD LIST is displayed and control returns to the CMS command environment with a return code of 4 in register 15.		NOLDL	
If the load list CNTRL file is not found, the message NO CONTROL FILE is displayed and control returns to the CMS command environment with a return code of 2 in register 15.		NOCTR	
3 The first record of the control file is read and the class on the macro library record is saved.	VMFLOAD	DINITA	
The rest of the control file is read. The control records are chained together in the proper hierarchy.		RDCTR	
If an error occurs while reading the control file, the message ERROR IN CONTROL FILE is displayed and control returns to the CMS command environment with a return code of 3 in register 15.		BDCTR	
4 The punch is set to spool continuously. The load list EXEC procedure is invoked by an SVC 202. The text files are punched in the order specified in the load list.	VMFLOAD	DINITB	
The resident nucleus modules are loaded first and the pageable modules follow. The DMKLD00E (nucleus loader) resident nucleus module must be loaded first and followed by DMKPSA. The DMKCPE module must be the last resident nucleus module loaded. The pageable nucleus modules are ordered so that they efficiently utilize page frames. The DMKSAV module must be loaded last. When the filename and filetype are both specified, that specific file is searched for and punched, if found. If the file is not found, it is skipped, the message filename filetype NOT FOUND is displayed, and processing continues with the next item in the load list.		NOFILE	
When only the filename is specified, the specified control file is used to		FNDM	
search for the highest level text file		DINITD	
available. The first text file located is punched. If the search ends before		SRTXT	
a text file is found, the "filename TEXT" file is punched if it exists. If the file is not found, it is skipped, the message filename filetype NOT FOUND is displayed, and processing continues with the next item in the load list.		NOFILE	
This process continues until every item in the load list is processed.			
5 At this point, the text decks are loaded in the proper sequence in the specified reader. All files not found were identified by messages to the terminal. The message SYSTEM LOAD DECK COMPLETE is displayed. The punch is set to stop spooling and is then closed.	VMFLOAD	ENDL	
Control returns to the CMS command environment.		RETERR	

Diagram 7-12. The Nucleus Load Program



1 Perform initial checking.

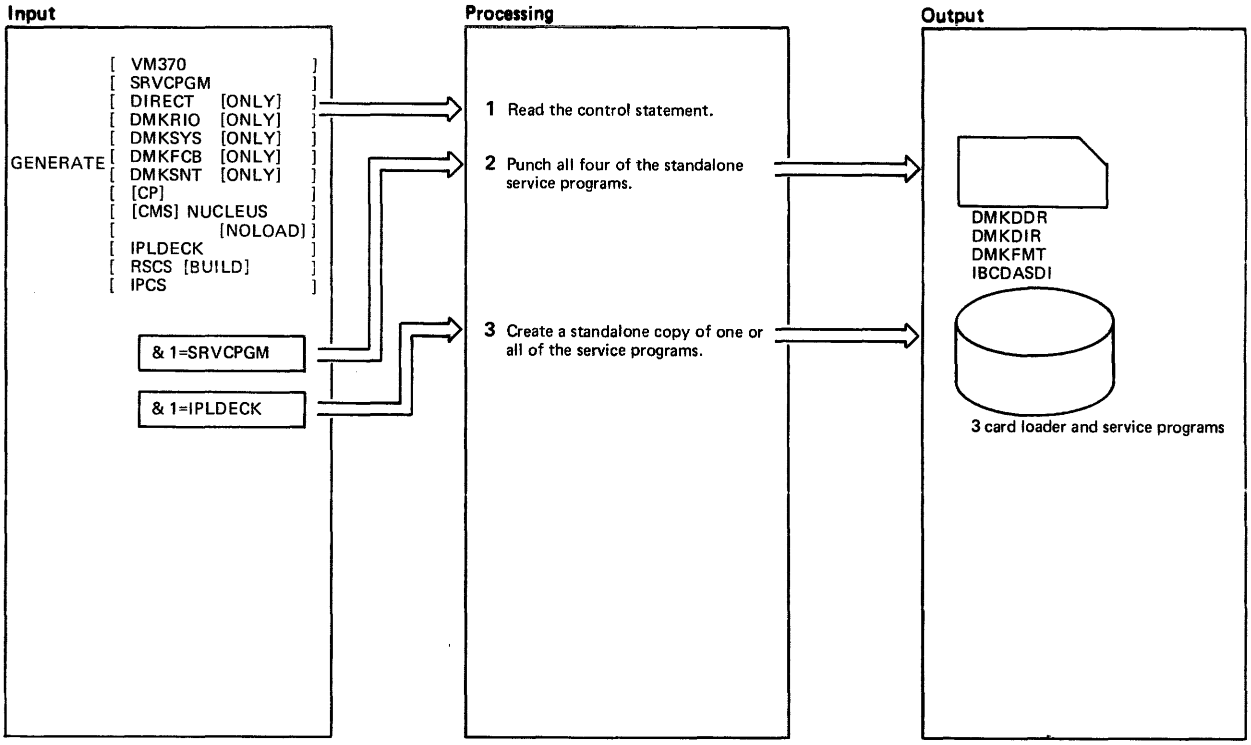
2 Print control file cntrlname CNTRL. Erase NEWMAC MACLIB and NEWMAC COPY.

3 Update macro and copy files. Add them to NEWMAC MACLIB. Print new members.

4 Erase any existing libname MACLIB. Rename NEWMAC MACLIB to libname MACLIB.

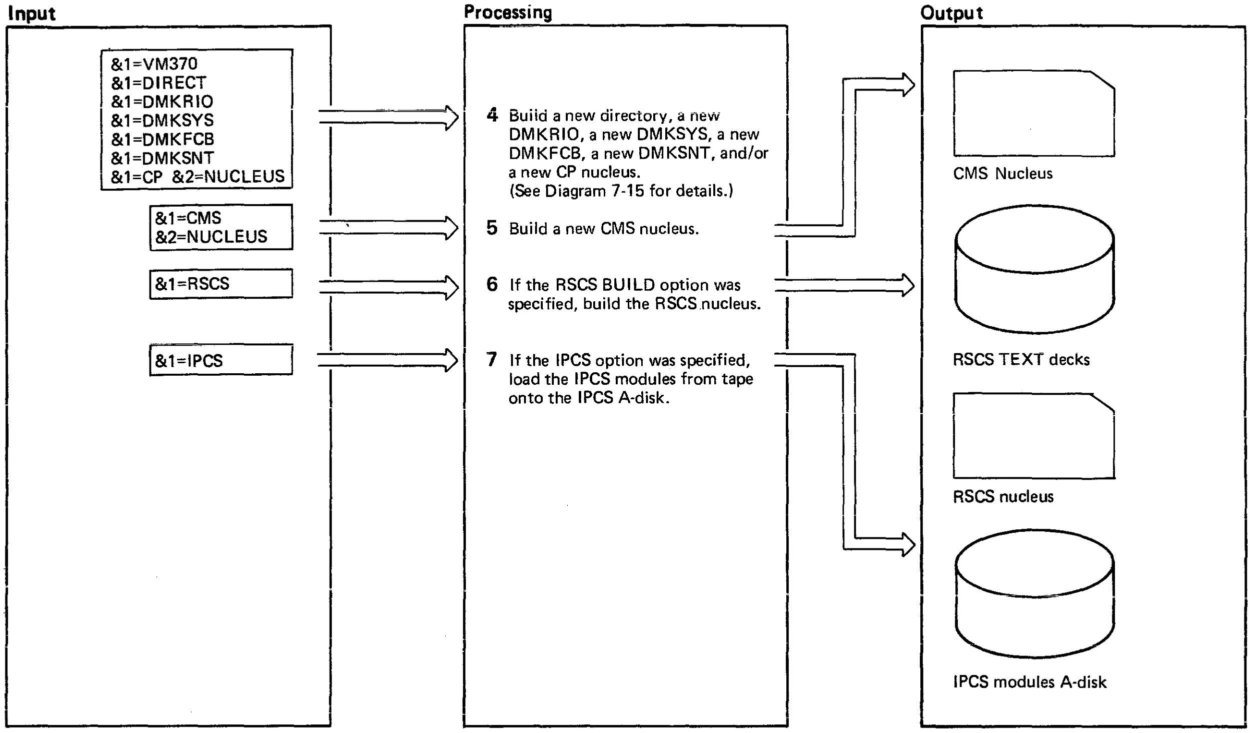
Notes	Module	Label	Ref
1 If a list of the members to be put in the macro library (maclibname EXEC) is not found, the message maclibname EXEC NOT FOUND is displayed and control returns to CMS with a return code of 101.	VMFMAC	-ASGN	
If the file containing the updates is not found, the message cntrlname CNTRL NOT FOUND is displayed and control returns to CMS with a return code of 102.		-STCTL	
2 The control file cntrlname CNTRL is printed. The files NEWMAC MACLIB and NEWMAC COPY are erased.	VMFMAC	-STKL	
3 If a macro or copy file is not found, the message *** filename COPY OR MACRO NOT FOUND *** is displayed. The final return code is set to 104 and processing continues with the next member.	VMFMAC	-AREAD	
The UPDATE command is issued for each macro or copy file.		-MACUP	
If an error occurs, the message *** ERRORS UPDATING membername membertype *** membername membertype NOT INCLUDED IN MACLIB is displayed on the terminal, the files membername UPDATES and membername membertype are printed. The final return code is set to 105 and processing continues with the next member.		-UPDERR	
If the update procedure is successful, VMFDATE is executed to date stamp the file, and the member is added to the NEWMAC MACLIB. The new member is printed. To maintain a history of the updates that were applied, a line is added to NEWMAC COPY, a dummy copy file.		-MACUP	
4 After all macro and copy files have been processed, the NEWMAC COPY file is renamed to libname COPY and added to NEWMAC MACLIB. Any existing libname MACLIB file is erased and the NEWMAC MACLIB is renamed to libname MACLIB.		-RENEWCO	
If the update procedure is unsuccessful, the message DUE TO PREVIOUS ERRORS, THE RESULT OF THIS MACLIB BUILD IS CALLED 'NEWMAC MACLIB' libname MACLIB HAS NOT BEEN REPLACED is displayed at the terminal and a return is made to CMS with the final return code as previously described.		-ERR2	

Diagram 7-13. VMFMAC--The Macro Library Creation Procedure



Notes	Module	Label	Ref
1 If an invalid operand is read, one of the following messages is displayed. GENERATE xxxxxxxx–INVALID OPERAND NUCLEUS OPTION–(CP/CMS) NOT SPECIFIED	GENERATE	-RMOVE	
2 The message: THE FOLLOWING STANDALONE SERVICE PROGRAMS ARE BEING PUNCHED ** FORMAT – DIRECT – DUMP/RESTORE – IBCDASDI ** is displayed, then each of the four standalone service programs is punched and the messages: PUNCHING 'IPL FMT' ****** PUNCHING 'IPL DIR' ****** PUNCHING 'IPL DDR' ****** PUNCHING 'IPL IBCDASDI' ***** are displayed. If the directory does not exist, the program terminates. Otherwise, the message: PRINT COPY OF RELEASE3 DIRECT? –RESPOND (YES/NO) is displayed. If the response is no, the program terminates. Otherwise a summary of the GENERATE EXEC procedure is displayed. See Figure 7-2. Then, the GENERATE EXEC prints the directory, DMKSYS, DMKSNT, and DMKFCB. Another message DO YOU WISH TO HAVE A COPY OF DMKSNT, DMKSYS, DMKFCB, AND RELEASE3 DIRECT PUNCHED TO CARDS? –RESPOND (YES/NO): is displayed. If the response is no, the program terminates. Otherwise, DMKSNT, DMKSYS, DMKFCB, and RELEASE3 DIRECT are punched, the following messages are displayed: PUNCHING 'DMKSNT ASSEMBLE' ****** PUNCHING 'DMKSYS ASSEMBLE' ****** PUNCHING 'DMKFCB ASSEMBLE' ****** PUNCHING 'RELEASE3 DIRECT' ****** and processing ends.	GENERATE	-SRVC	
3 The instructions: ENTER THOSE DECKS TO BE GENERATED (DDR/DIR/FMT/ ALL): ENTER TARGET DISK ADDRESS: are displayed on the terminal. If the target disk address entered is 190, the filemode is set to A. Otherwise, it is set to B. A 3-card loader is placed in front of each of the service programs specified and the service program is copied. At successful completion, one or all of the following messages 'IPL DIR A1' – CREATED 'IPL DDR A1' – CREATED 'IPL FMT A1' – CREATED is displayed and the GENERATE procedure ends. If an error occurs, an error message: xxxxxxxx – INVALID OPERAND. ERROR ON ACCESS OF DISK (xxxxxxx) ERROR WRITING OR BUILDING 'IPL [DIR/DDR/FMT] A' DISK A (xxxxxx) – READ ONLY. is displayed and processing ends.	GENERATE	-IPLGEN -GTDISK	

Diagram 7-14. The GENERATE Procedure (Part 1 of 2)



Input

&1=VM370
&1=DIRECT
&1=DMKRIO
&1=DMKSYS
&1=DMKFCB
&1=DMKSNT
&1=CP &2=NUCLEUS

&1=CMS
&2=NUCLEUS

&1=RSCS

&1=IPCS

Processing

4 Build a new directory, a new DMKRIO, a new DMKSYS, a new DMKFCB, a new DMKSNT, and/or a new CP nucleus. (See Diagram 7-15 for details.)

5 Build a new CMS nucleus.

6 If the RSCS BUILD option was specified, build the RSCS nucleus.

7 If the IPCS option was specified, load the IPCS modules from tape onto the IPCS A-disk.

Output

CMS Nucleus

RSCS TEXT decks

RSCS nucleus

IPCS modules A-disk

Notes	Module	Label	Ref
4 See Diagram 7-15.	GENERATE	-VM370 -NUCLEUS	
5 The load list name is set to CMSLOAD and the control file name is set. Then VMFLOAD loads the CMS modules and the nucleus is created. A final question: DO YOU WANT A CARD IMAGE COPY OF THE CMS NUCLEUS LOAD DECK AS A DISK FILE – RESPOND (YES/NO) is displayed and the processing ends. If the response is yes, the CMS nucleus is written on disk, and the message CMS NUCLEUS LOAD DECK EXISTS ON DISK AS 'CMSNUC NUCLEUS A1' is displayed. If NOLOAD was specified, a copy of the CMS nucleus is left in the virtual card reader. Otherwise, it is loaded.	GENERATE	-CMS -BUILD -DMS	
6 The question DO YOU WISH TO BUILD RSCS SYSTEM – RESPOND (YES/NO) is displayed. If the user replies "yes", he is prompted:	GENERATE	-RSCS	
ENTER RSCS SYSTEM DISK LINK PARAMETERS: USERID VADDR1 VADDR2 The disk the user specifies is linked to and accessed as the A-disk. The AXS, LAX, NPT, and SML modules are copies onto the A-disk. The message TRANSFERRING 'RSCS' DISK RESIDENT TEXT . . . is displayed.		-RSCSBLD	
Then VMFLOAD builds the nucleus. The message WHEN THE NEW RSCS SYSTEM IS BUILT, ISSUE: 'CLOSE PRT'. . . (PRINTS THE LOAD MAP) is displayed, the virtual card reader is IPLed, and processing ends.		-BLDSYS	
7 The message *** IPCS SYSTEM BUILD *** *** IS THE CURRENT A-DISK TO BE THE IPCS A-DISK? RESPOND (YES/NO):	GENERATE	-IPCS -IPCSRP	
is displayed. If the user replies "yes", the IPCS modules are loaded from tape onto the A-disk.		-IPCSYES	
If the user replies "no", he is prompted: ENTER IPCS USERID		-IPCSRU	
ENTER IPCS A-DISK ADDRESS, LINK ADDRESS, AND WRITE PASSWORD The disk the user specifies is linked to and accessed as the A-disk, and the IPCS modules are loaded onto it from tape. If errors occur, error messages are displayed.		-IPCSGD	
When loading is completed, the message *** IPCS BUILD COMPLETE *** is displayed and processing ends.		-IPCSYES	

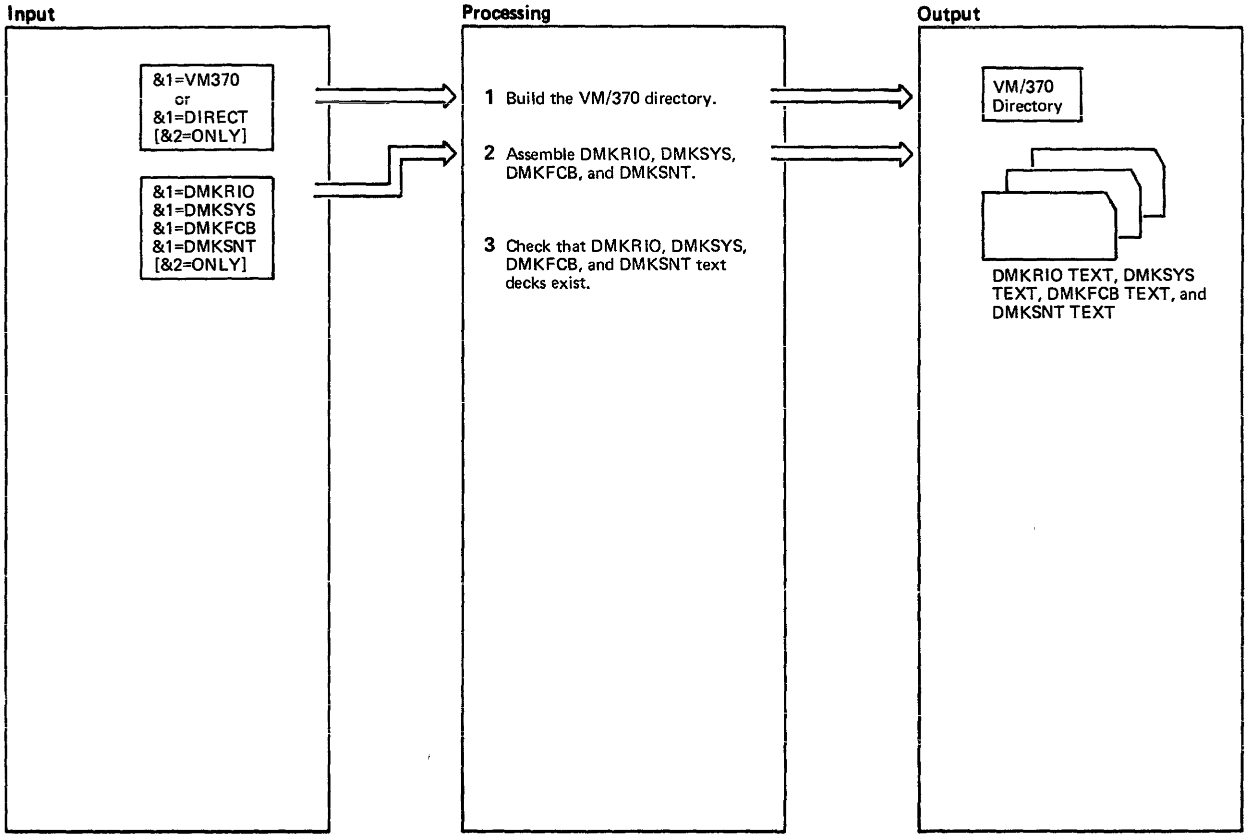
Diagram 7-14. The GENERATE Procedure (Part 2 of 2)

```
A SAMPLE DIRECTORY IS BEING PRINTED TO AID YOU.
IT SHOWS WHERE THE VIRTUAL DISKS ARE LOCATED ON 'CPV3L0'
YOU MAY USE THESE MINIDISKS FOR OTHER VIRTUAL MACHINES,
IN PARTICULAR THE CMS SYSTEM DISK (MAINT 190) ANC
THE CP STAGING AREA DISK (MAINT 194)
INCLUDED IN THIS DIRECTORY IS THE USERID: MAINT
WHICH WILL BE USED FOR FUTURE SUPPORT OF THE SYSTEM.
THIS USERID SHOULD BE INCLUDED IN THE DIRECTORY YOU BUILD
FOR YOUR FLOOR USE.
** CAUTION ** IF YOU DESTROY USER MAINT'S AREAS, IT WILL BE
              NECESSARY TO RE-BUILD THE ENTIRE SYSTEM.

A SAMPLE OF DMKSYS, DMKFCB, AND DMKSNT ASSEMBLE ARE ALSO BEING
PRINTEC TO AID YOU. THIS SAMPLE DMKSNT IS BASED ON THE
INFORMATION INCLUDED IN THE SAMPLE DMKSYS AS WELL AS THE
EXAMPLE ALLOCATIONS FOR VMREL3 PROVIDED IN THE SYSGEN GUIDE.
A COPY OF THIS DMKSNT MODULE HAS BEEN INCLUDED IN THE CP NUCLEUS,
SUCH THAT IF ONE USES THE INCLUDED DMKSYS AND THE
SAMPLE ALLOCATION PROVIDED IN THE SYSTEM GENERATION GUICE,
HE WILL BE ABLE TO SAVE HIS CMS SYSTEM UPON COMPLETION
OF THE SYSTEM GENERATION PROCEDURE. A COPY OF DMKFCB HAS BEEN
INCLUDED IN THE NUCLEUS AND NEED NOT BE RE-ASSEMELED FOR
SYSTEM GENERATION. IT HAS BEEN INCLUDED FOR THE USER WHO WOULD LIKE
TO MODIFY OR ADD TO THE EXISTING BUFFER LOAD.

NOTE:  IF THE USER WISHES TO MODIFY THE SAMPLE DMKSNT AND/OR DMKFCB
HE MAY INCLUDE THE UPDATED SOURCE WITH THE SOURCE INCLUCED UNDER
THE OPTION 'GENERATE VM370', OF THE SYSTEM GENERATION PROCEDURE.
IF PRESENT, IT WILL AUTOMATICALLY BE ASSEMBLED ANC INCLUDED IN THE
NEW CP NUCLEUS.
```

Figure 7-2. GENERATE Introductory Message



Input

&1=VM370
or
&1=DIRECT
[&2=ONLY]

&1=DMKRIO
&1=DMKSYS
&1=DMKFCB
&1=DMKSNT
[&2=ONLY]

Processing

1 Build the VM/370 directory.

2 Assemble DMKRIO, DMKSYS, DMKFCB, and DMKSNT.

3 Check that DMKRIO, DMKSYS, DMKFCB, and DMKSNT text decks exist.

Output

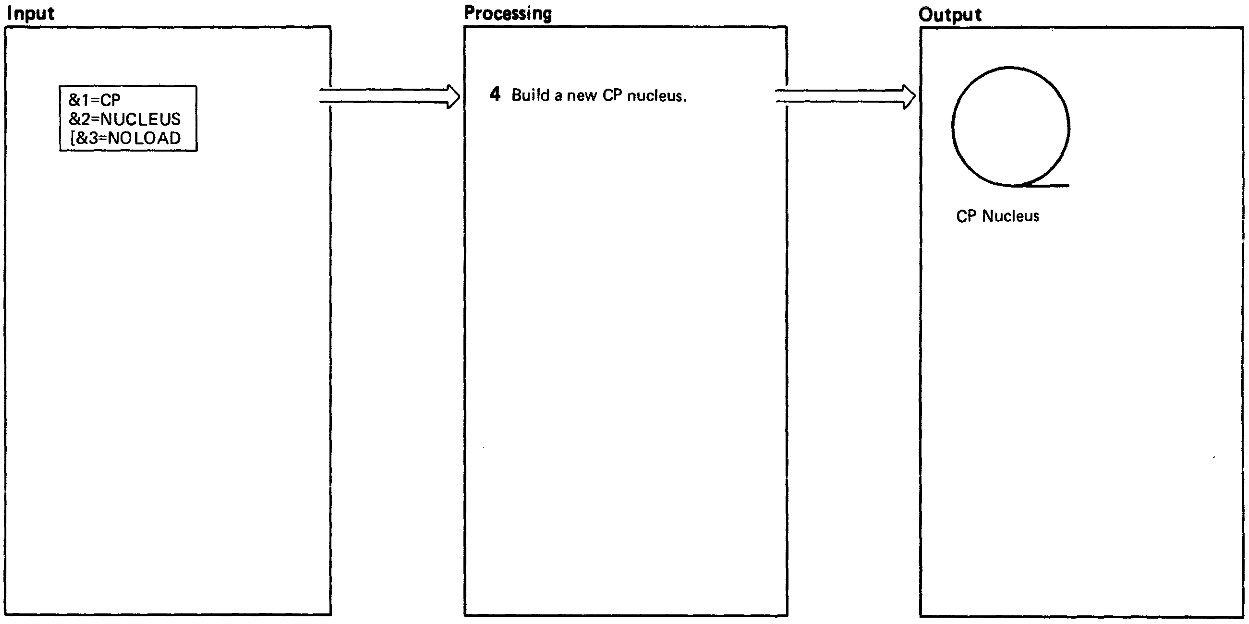
VM/370
Directory

DMKRIO TEXT, DMKSYS TEXT, DMKFCB TEXT, and DMKSNT TEXT

Notes	Module	Label	Ref
1 If &1=VM370 or DIRECT, the directory is built and the message THE **DIRECTORY** HAS BEEN BUILT is displayed. If &2=ONLY, processing terminates. Otherwise, processing continues with step 2.	GENERATE	-VM370 -DIRECT	
If an error occurs, the message CORRECT THE DIRECTORY CARDS AND RELOAD THE CARD READER RESPOND WITH: GENERATE DIRECT is displayed and processing terminates.		-BADDRCT	
2 The DMKRIO, DMKSYS, DMKFCB, and DMKSNT modules are assembled using the VMFASM procedure. If &2=ONLY, processing terminates. Otherwise, processing continues with step 3.	GENERATE	-RUN	
If an error occurs, the message CORRECT THE [DMKRIO/DMKSYS/ DMKFCB/DMKSNT] FILE AND RELOAD THE CARD READER RESPOND WITH: GENERATE [DMKRIO/DMKSYS/DMKFCB/ DMKSNT] is displayed and the processing stops.		-ASMERR	

Notes	Module	Label	Ref
3 The CMS STATE command is issued to see that the DMKRIO, DMKSYS, DMKFCB, and DMKSNT text decks exist.	GENERATE	-LPEND	

Diagram 7-15. Generating a CP System (Part 1 of 2)



Notes	Module	Label	Ref
4 The load list name is set to CPLOAD and the control file &CTL is set. The message VIRTUAL=REAL OPTION REQUIRED (YES, NO): is displayed. If you respond yes, the message STORAGE SIZE OF VIRT=REAL <MINIMUM IS 32K> is displayed. If you enter a storage size that is not a multiple of 4K, the message ** SIZE ROUNDED UP TO NEXT HIGHER 4K BOUNDARY ** is displayed. The messages STORAGE SIZE FOR VIRTUAL=REALnnnnK IS THE ABOVE ENTRY CORRECT (YES, NO): are displayed. If you respond no, the process is repeated. If you respond yes, the value of the storage size is set in the DMKSLC TEXT file. If an error occurs, the message ERROR WHILE WRITING "DMKSLC TEXT" FILE is displayed, and the system will be built without a virtual=real area. Otherwise, the load list name is set to VRLOAD.	GENERATE	-CP	
The VMFLOAD program is invoked to load the modules and the CP nucleus is then written on tape. The message IPLABLE NUCLEUS NOW ON TAPE ****** is displayed. If the CP system was built without a virtual=real area, the message WHEN 'NUCLEUS LOADED ON xxxxxx' IS TYPED, ISSUE 'CLOSE PRT' TO GET THE CPLOAD MAP. WHEN PRINTING IS COMPLETE, SHUTDOWN THE SYSTEM AND IPL THE NEW SYSRES VOLUME. is displayed on the terminal. If &3=NOLOAD, the tape is not loaded. Otherwise the tape containing the CP nucleus is loaded. Processing ends.		-BUILD	
If the CP system does contain a virtual=real area, however, the tape containing the CP nucleus is not loaded, and the following message is displayed: TO LOAD THE CP NUCLEUS JUST CREATED, SHUTDOWN THE SYSTEM AND THEN IPL THE TAPE. THE CPLOAD MAP WILL AUTOMATICALLY BE PRINTED AT THE PRINTER WHOSE ADDRESS IS '00E'. IF THERE IS NO PRINTER AT THIS ADDRESS THE LOAD MAP WILL BE PRINTED AT THE FIRST PRINTER CAUSING AN INTERRUPT, (IE. NOT READY TO READY SEQUENCE). ONCE THE NUCLEUS HAS BEEN LOADED, YOU MAY IPL YOUR NEW CP SYSTEM RESIDENCE VOLUME. NOTE: THERE MUST BE ENOUGH STORAGE ON THE SYSTEM (VIRTUAL OR REAL), TO CONTAIN THE VIRT=REAL AREA AND THE CP NUCLEUS.		-REALIPL	
Processing ends. If an error occurs while writing the CP nucleus to tape, one or more of the following messages appears: TAPE (182) – NOT READY OR NOT ATTACHED HIT RETURN WHEN READY OR 'EXIT': ERROR BUILDING xxxxxxxx NUCLEUS. ERROR WRITING CP NUCLEUS TO TAPE.			

Diagram 7-15. Generating a CP System (Part 2 of 2)

Program Organization

The VM/370 procedures for generating and updating VM/370 consist of three EXEC procedures (VMFASM, VMFMAC, and GENERATE) and three modules (VMFDATE, DMSUPD, and VMFLOAD).

The Assembler language update procedure consists of the VMFASM EXEC procedure and two modules (VMFDATE and DMSUPD). The VMFASM EXEC procedure sets up for the assembly by calling DMSUPD to create the update control file. There is an entry in the VMCNTRL file for each update control and auxiliary update file. The VMCNTRL identifies the updates applied to the original assembler program and the date and time they were applied.

The Assembler language update procedure calls the VMFDATE program. The MACLIBs needed are then included in the VMCNTRL file.

The nucleus loader procedure consists of a program (VMFLOAD) and an EXEC procedure. Although the DMSUPD update program is not used, the control file that it creates may be used. The LOADER EXEC procedure lists the nucleus modules in the order they are to be loaded. The list includes the filename of each module and may optionally include the update level. If the update level is not specified, the control file created by DMSUPD is used to locate the highest level update available, and that level of the module is loaded.

When nucleus modules are updated and loaded, it is often necessary to create a new macro library. The level of macro library needed for each updated module is recorded in the VMCNTRL file created by the VMFDATE module. The VMFMAC EXEC procedure creates a new macro library.

The GENERATE EXEC procedure reassembles the DMKRIO, DMKSYS, DMKFCB, and DMKSNT modules by using the VMFASM EXEC procedure. It loads the CP, CMS, or RSCS nucleus using the VMFLOAD program. In addition, it can build a new VM/370 directory, punch the standalone service programs, or make a self-relocating copy of the service programs, or load the IPCS modules from tape onto the IPCS A-disk.

Directory

Four label directories are provided. Figure 7-3 is the label directory for the Assembler update function, including labels from:

- The VMFASM EXEC procedure.
- The DMSUPD update program.
- The VMFDATE control file program.

Figure 7-4 is the label directory for the nucleus load program, VMFLOAD.

Figure 7-5 is the label directory for the VMFMAC EXEC procedure, which creates and updates the macro library.

Figure 7-6 is the label directory for the GENERATE EXEC procedure, which creates new service program decks, builds a new CP, CMS, or RSCS nucleus, or loads the IPCS modules from tape onto the IPCS A-disk.

ASSEMBLE UPDATE PROCEDURE

Label	Module or Procedure	Diagram	Description
-ASMP	VMFASM	7-3	Assumes default options for Assembler.
AUXFINT	DMSUPD	7-7	Closes the auxiliary file when it is completely processed.
AUXREAD	DMSUPD	7-7	Reads auxiliary file from the bottom up.
BADAUXC	DMSUPD	7-7	Processing when invalid card found in auxiliary file.
BADCTLC	DMSUPD	7-7	Abnormally terminates when an invalid control card is encountered.
-COMB	VMFASM	7-3	Saves the new text file, original ASSEMBLE file, and UPDTxxxx files.
CORBUST	DMSUPD	7-10	Insufficient storage to complete update.
CTLDONE	DMSUPD	7-7	Closes the control file once it is processed.
CTLGETM	DMSUPD	7-7	Searches for first control card.
CTLGOT1	DMSUPD	7-7	Checks that auxiliary file exists.
CTLIPTF	DMSUPD	7-7	Checks that PTF file exists.
CTLMULT	DMSUPD	7-5 7-7	Multiple update processing.
CTLOCUP	DMSUPD	7-7	Checks that update file exists.
CTLREAD	DMSUPD	7-7	Reads the control file from the bottom up.
CTLUMSG	DMSUPD	7-7	Updates the UPDATES file.
CTLUMSS	DMSUPD	7-7	Issues the short update message.
DELTINE	DMSUPD	7-9	Deletes cards from the source file.
DMSUPD	DMSUPD	7-6	Entry to update program.
-DTF	VMFASM	7-3	Stacks control file in printer.
ERMACS	DMSUPD	7-7	Processing when MACS card invalid or missing.
ERSC	DMSUPD	7-6	Processing when STK option specified without CTL option.
EXCESIV	DMSUPD	7-6	Error exit when too many parameters are specified.
-EXIT	VMFASM	7-3	Erases intermediate files and returns to CMS.
FCTDELT	DMSUPD	7-9	Checks the delete control card for validity.
FCTINST	DMSUPD	7-9 7-10	Checks the validity of the insert control card.
FCTREPL	DMSUPD	7-10	Checks the validity of the replace control card.
FCTRSEQ	DMSUPD	7-9	Checks the resequence control card.
-FUPD	VMFASM	7-2 7-3	Assembles the updated program.
IMPLICIT	DMSUPD	7-8	Update processing will be done using disk.
INSEQW	DMSUPD	7-10	Processing when sequence errors occur in input file.
INSLOOP	DMSUPD	7-10	Inserts cards from the source file.
INVCHAR	DMSUPD	7-9	Processing for invalid character in sequence field.

Figure 7-3. The Assembler Update Procedure Label Directory (Part 1 of 2)

Label	Module or Procedure	Diagram	Description
INVOPTN	DMSUPD	7-6	Error exit when an unrecognizable option is encountered.
INVUPCD	DMSUPD	7-10	Processing for invalid update file control card.
LOCTUPD	DMSUPD	7-5	Checks that a single update file exists.
NOERASE	DMSUPD	7-5	Checks that the control file exists.
NOFILE	DMSUPD	7-5	Processing when the source input file is not found.
NOFILEW	DMSUPD	7-7	Processing when PTF file not found.
NOFNAME	DMSUPD	7-6	Error exit when no operands were entered.
NOUPDATS	DMSUPD	7-5	Abnormally terminates when update file specified but not found.
OPTCONF	DMSUPD	7-6	Abnormally terminates when conflicting options specified.
OPTDUP	DMSUPD	7-6	Abnormally terminates when the same option is specified more than once.
PROCESS	DMSUPD	7-5	Checks if the update and source input files already exist.
RETRD	DMSUPD	7-11	Creates disk output file from the in-storage updated file.
RETR001	DMSUPD	7-11	Closes and renames the created output disk file.
RETURN	DMSUPD		Checks RETCODE for indication of warning messages.
RSEQDEF	DMSUPD	7-9	Sets the sequencing to 5 or 8 characters.
RSEQERR	DMSUPD	7-9	Issues DMSUPD184W message.
RSEQFIN	DMSUPD	7-9	Sets up for resequencing.
SINGUPD	DMSUPD	7-5 7-9	Applies a single update.
SMALLCOR	DMSUPD	7-8	Insufficient storage to begin update.
-STCTL	VMFASM	7-2	Checks for CNTRL file.
-STSYS	VMFASM	7-2	Checks for the ASSEMBLE file.
TEST	VMFDATE	7-4	Checks for the input file.
UPDREAD	DMSUPD	7-9	Reads control cards.
UPDSERR	DMSUPD	7-10	Issues DMSUPD186W message.
VMFDATE	VMFDATE	7-4	Creates the UPDATES file.
WOVF	DMSUPD	7-10	Issues DMSUPD176W message.
WRETURN	DMSUPD	7-5	Issues DMSUPD177I message.
XDELE	DMSUPD	7-9	Deletes line from storage.
XWRITE	DMSUPD	7-10	Inserts line into storage.
ZERSEQ	DMSUPD	7-9	Issues DMSUPD182W message.

Figure 7-3. The Assembler Update Procedure Label Directory (Part 2 of 2)

VMFLOAD PROGRAM

Label	Module or Procedure	Diagram	Description
BDCTR	VMFLOAD	7-12	Error exit when error occurs while reading control file.
DINITA	VMFLOAD	7-12	Reads the MACS record from control file.
DINITB	VMFLOAD	7-12	Punches text files.
DINITD	VMFLOAD	7-12	Punches the highest level update available.
ENDL	VMFLOAD	7-12	Closes punch and returns to CMS.
FNDM	VMFLOAD	7-12	Searches for file specified in control file.
NOCTR	VMFLOAD	7-12	Error exit when control file not found.
NOFILE	VMFLOAD	7-12	Skips the files that are not found.
NOLDL	VMFLOAD	7-12	Error exit when loadlist EXEC procedure is not found.
RDCTR	VMFLOAD	7-12	Reads the control file.
RETERR	VMFLOAD	7-12	Exits to CMS.
SRTXT	VMFLOAD	7-12	Punches the TEXT file if update level is not found.
VMFLOAD	VMFLOAD	7-12	Entry for load list program.

Figure 7-4. The VMFLOAD Program Label Directory

VMFMAC PROCEDURE

Label	Module or Procedure	Diagram	Description
-AREAD	VMFMAC	7-13	Checks that each macro or copy file listed in the 'maclibname EXEC' file exists.
-ASGN	VMFMAC	7-13	Checks that the 'maclibname EXEC' file exists.
-ERR2	VMFMAC	7-13	Prints error message if entire update prccedure is not successful.
-MACUP	VMFMAC	7-13	Updates the macro or copy files and puts them in the new macro library.
-RENEWCO	VMFMAC	7-13	Renames existing NEWMAC COPY and NEWMAC MACLIB files.
-STCTL	VMFMAC	7-13	Checks that the 'cntrlname CNTRL' file exists.
-STKL	VMFMAC	7-13	Prints the control file.
-UPDERR	VMFMAC	7-13	Prints error message if error occurs during updating.

Figure 7-5. The VMFMAC Procedure Label Directory

Label	Module or Procedure	Diagram	Description
-ASMERR	GENERATE	7-15	Sends an error message and exits when an error is encountered while assembling DMKRIO, DMKSYS, DMKFCB or DMKSNT.
-BADDRCT	GENERATE	7-15	Sends an error message and exits when an error is encountered while building the directory.
-BLDSYS	GENERATE	7-14	Invokes the VMFLOAD program to build the RSCS nucleus.
-BUILD	GENERATE	7-14 7-15	Invokes the VMFLOAD program.
-CMS	GENERATE	7-14	Sets the correct load list and control file for the CMS nucleus load.
-CP	GENERATE	7-15	Sets the correct load list and control file for the CP nucleus load. Sets up virtual=real area if desired.
-DIRECT	GENERATE	7-15	Builds the VM/370 directory.
-DMS	GENERATE	7-14	Saves a copy of the CMS nucleus on disk.
-IPCS -IPCSRP	GENERATE	7-14	Prompts the user to verify that the current A-disk is to be used as the IPCS A-disk.
-IPCSRU	GENERATE	7-14	Prompts the user for the IPCS userid.
-IPCSGD	GENERATE	7-14	Prompts the user for the IPCS A-disk link parameters.
-IPCSYES	GENERATE	7-14	Loads the IPCS modules from tape onto the IPCS A-disk.
-IPLGEN	GENERATE	7-14	Creates a standalone copy of one or all of the service programs.
-LPEND	GENERATE	7-15	Checks whether the DMKRIO, DMKFCB, DMKSYS, or DMKSNT TEXT decks exist.
-NUCLEUS	GENERATE	7-14	Builds a new CP or CMS nucleus.
-RSCS	GENERATE	7-14	Prompts the user to build the RSCS system.
-RSCSBLD	GENERATE	7-14	Writes the RSCS text decks on disk.
-RUN	GENERATE	7-15	Assembles the DMKRIO, DMKSYS, DMKFCB, and DMKSNT decks.
-SRVC	GENERATE	7-14	Punches the standalone service programs.
-VM370	GENERATE	7-14 7-15	Main processing routine for building new CP modules or VM/370 directory.

Figure 7-6. The GENERATE Procedure Label Directory

Diagnostic Aids

The following figures list all the messages issued by the modules and EXEC procedures that create and update the VM/370 system. Figure 7-7 lists all the messages issued by the VMFASM EXEC procedure, Figure 7-8 lists the messages issued by the DMSUPD module, Figure 7-9 lists the messages issued by the VMFLOAD program, Figure 7-10 lists the messages issued by the VMFMAC procedure, and Figure 7-11 lists the messages issued by the GENERATE procedure. The label cf the issuing routine and the diagram (if any) describing that routine are included.

VMFASM PROCEDURE

Label	Diagram	Message Text
-FUPD	7-3	***ERROR UPDATING filename***
-ASMP	7-3	ASMBLING filename (options...)
-DTF	7-3	***ERROR ASMBLING filename***
-DTF	7-3	***NO TEXT FOR filename***
-COMB	7-3	filename $\left\{\begin{array}{l}\text{TEXT}\\\text{TXTxxxxx}\end{array}\right\}$ CREATED

Figure 7-7. VMFASM Messages

Message Code	Label	Diagram	Return Code or Severity	Message Text
DMSUPD001E	NOFNAME	7-6	24	NO FILENAME SPECIFIED
DMSUPD002E	NOFILE	7-5	28	FILE 'fn ft fm' NOT FOUND
DMSUPD003E	INVOPTN	7-6	24	INVALID OPTION 'option'
DMSUPD007E	FMTERR		32	FILE 'fn ft fm' NOT FIXED, 80 CHAR. RECORDS
DMSUPD010W	INPFERR		12	PREMATURE EOF ON FILE 'fn ft fm' --SEQ NUMBER '........' NOT FOUND
DMSUPD024E	PROCESS ERCMSUT	7-5	24	FILE 'UPDATE CMSUT1 fm' ALREADY EXISTS
DMSUPD037E	PROCESS ERRW	7-5	36	DISK 'A' IS READ/ONLY
DMSUPD048E	BADMODE		24	INVALID MODE 'mode'
DMSUPD065E	OPTDUP	7-6	24	'option' OPTION SPECIFIED TWICE
DMSUPD066E	OPTCONF	7-6	24	'option' AND 'option' ARE CONFLICTING OPTIONS
DMSUPD069E	NOTACCER		32	DISK 'A' NOT ACCESSED
DMSUPD070E	EXCESIV	7-6	24	INVALID PARAMETER 'param'
DMSUPD104S	INPERR		100	ERROR 'nn' READING FILE 'fn ft fm' FROM DISK
DMSUPD105S	OUTERR		100	ERROR 'nn' WRITING FILE 'fn ft fm' ON DISK
DMSUPD174W	INSLOOP PASSW	7-10	8	SEQUENCE ERROR INTRODUCED IN OUTPUT FILE: 'xxx' TO 'xxx'
DMSUPD176W	WOVF	7-10	8	SEQUENCING OVERFLOW FOLLOWING SEQUENCE NUMBER 'xxx'
DMSUPD177I	WRETURN	7-5	-	WARNING MESSAGES ISSUED (SEVERITY = nn). <'REP' OPTION IGNORED>
DMSUPD178I	CTLUMSG	7-7	-	UPDATING 'fn ft fm' WITH 'fn ft fm'
DMSUPD179E	ERMACS	7-7	32	MISSING OR DUPLICATE 'MACS' CARD IN CONTROL FILE 'fn ft fm'
DMSUPD180W	NOFILEW	7-7	12	MISSING PTF FILE 'fn ft fm'
DMSUPD181E	NOUPDATS	7-5	40	NO UPDATE FILES WERE FOUND
DMSUPD182W	ZERSEQ	7-9	8	SEQUENCE INCREMENT IS ZERO
DMSUPD183E	BADCTLC BADAUXC	7-7	32	INVALID {CONTROL | AUX} FILE CONTROL CARD
DMSUPD184W	RSEQERR	7-9	12	'./S' NOT FIRST CARD IN UPDATE FILE--IGNORED
DMSUPD185W	INVCHAR	7-9	12	INVALID CHAR IN SEQUENCE FIELD 'xxxxxx'
DMSUPD186W	UPDSERR	7-10	12	SEQUENCE NUMBER 'xxx' NOT FOUND
DMSUPD187E	ERSC	7-6	24	OPTION 'STK' INVALID WITHOUT 'CTL'
DMSUPD208W	UPDREAD INVUPCD	7-9 7-10	12	INVALID UPDATE FILE CONTROL CARD
DMSUPD210W	INSEQW	7-10	4	INPUT FILE SEQUENCE ERROR: 'xxx' TO 'xxx'
DMSUPD299E	CORBUST	7-10	40	INSUFFICIENT STORAGE TO COMPLETE UPDATE
DMSUPD300E	SMALLCOR	7-7	40	INSUFFICIENT STORAGE TO BEGIN UPDATE
DMSUPD304I	IMPLICIT	7-7	-	UPDATE PROCESSING WILL BE DONE USING DISK.

Figure 7-8. DMSUPD Messages

VMFLOAD PROGRAM

Label	Diagram	Message Text
NOFILE	7-12	filename filetype NOT FOUND
BDCTR	7-12	ERROR IN CONTROL FILE
NOCTR	7-12	NO CONTROL FILE
NOLDL	7-12	NO LOAD LIST
ENDL	7-12	SYSTEM LOAD DECK COMPLETE

Figure 7-9. VMFLOAD Messages

VMFMAC PROCEDURE

Label	Diagram	Message Text
-ASGN	7-13	*** maclibname EXEC NOT FOUND ***
-STCTL	7-13	*** cntrlname CNTRL NOT FOUND ***
-AREAD	7-13	*** filename COPY OR MACRO NOT FOUND ***
-UPDERR	7-13	*** ERRORS UPDATING membername membertype *** membername membertype NOT INCLUDED IN MACLIB
-ERR2	7-13	DUE TO PREVIOUS ERRORS, THE RESULT OF THIS MACLIB BUILD IS CALLED 'NEWMAC MACLIB', libname MACLIB HAS NCT BEEN REPLACED

Figure 7-10. VMFMAC Messages

Label	Diagram	Message Text
	7-14	GENERATE xxxxxxxx--INVALID OPERAND.
-NUCLEUS	7-14	NUCLEUS OPTION -- (CP|CMS) NOT SPECIFIED.
-SRVC	7-14	THE FOLLOWING STANDALONE SERVICE PROGRAMS ARE BEING PUNCHED **FORMAT - DIRECT - DUMP/RESTORE - IBCDASDI**
	7-14	PUNCHING 'IPL FMT' ******
	7-14	PUNCHING 'IPL DIR' ******
	7-14	PUNCHING 'IPL DDR' ******
	7-14	PUNCHING 'IPL IBCDASDI' ******
	7-14	PRINT COPY OF RELEASE2 DIRECT? -- RESPOND (YES|NO)
	7-14	A SAMPLE DIRECTORY IS BEING PRINTED TO AID YOU. ...
	7-14	DO YOU WISH TO HAVE A COPY OF DMKSNT, DMKSYS, DMKFCB and RELEASE2 DIRECT PUNCHED TO CARDS? -- RESPOND (YES|NO)
	7-14	PUNCHING 'DMKSYS ASSEMBLE' ******
	7-14	PUNCHING 'DMKSNT ASSEMBLE' ******
	7-14	PUNCHING 'DMKFCB ASSEMBLE' ******
	7-14	PUNCHING RELEASE2 DIRECT ******
-IPLGEN	7-14	ENTER THOSE DECKS TO BE GENERATED (DDR|DIR|FMT|ALL): ENTER TARGET DISK ADDRESS:
	7-14	IPL 'DDR A1' -- CREATED
	7-14	IPL 'DIR A1' -- CREATED
	7-14	IPL 'FMT A1' -- CREATED
	7-14	xxxxxxxx -- INVALID OPERAND
	7-14	ERROR WRITING OR BUILDING 'IPL xxx A'.
	7-14	ERROR ON ACCESS OF DISK (xxxxxxxx)
		DISK A (xxxxxx) -- READ ONLY
-DMS	7-14	DO YOU WANT A CARD IMAGE OF THE NUCLEUS LOAD DECK AS A DISK FILE -- RESPOND (YES|NO)
	7-14	CMS NUCLEUS LOAD DECK EXISTS ON DISK AS 'CMSNUC NUCLEUS A1'.
-RSCS	7-14	DO YOU WISH TO BUILD RSCS SYSTEM -- RESPOND (YES|NO)
-RSCSBLD	7-14	ENTER RSCS SYSTEM DISK LINK PARAMETERS: USERID VADDR1 VADDR2
	7-14	MISSING PARAMETERS -- RE-ENTER
		ERROR LINKING TO userid vaddr1 AS vaddr2

Figure 7-11. GENERATE Messages (Part 1 of 3)

Label	Diagram	Message Text
	7-14	TRANSFERRING 'RSCS' DISK RESIDENT TEXT
-BLDSYS	7-14	WHEN THE NEW RSCS SYSTEM IS BUILT, ISSUE: 'CLOSE PRT'...(PRINTS THE LOAD MAP)
-ERRTAP	7-14	*** ERROR READING RSCS TAPE OR WRITING TO DISK 194 ***
-ERRTRANS	7-14	ERROR TRANSFERRING RSCS DISK RESIDENT TEXT FILE.
-ERRSYS	7-14	*** ERROR CREATING THE RSCS NUCLEUS ***
-ERRACC	7-14	ERROR ACCESSING SPECIFIED DISK
-IPCS	7-14	*** IPCS SYSTEM BUILD *** *** IS THE CURRENT A-DISK TO BE THE IPCS A-DISK? REMEMBER THE SYSOPR MACRO IN DMKSYS MUST SPECIFY THE IPCS USERID IN SYSDUMP=USERID. THIS USERID'S A-DISK MUST BE USED FOR THE IPCS BUILD.
-IPCSRP	7-14	RESPOND (YES OR NO):
-IPCSRU	7-14	ENTER IPCS USERID
-IPCSGD	7-14	ENTER IPCS A-DISK ADDRESS, LINK ADDRESS, AND WRITE PASSWORD
-IPCSYES	7-14	*** IPCS BUILD COMPLETE ***
-IPCSTE	7-14	*** ERROR LOADING IPCS MODULES FROM TAPE
-IPCSLE	7-14	*** ERROR LINKING userid vaddr
-DIRECT	7-15	THE **DIRECTORY** HAS BEEN BUILT
-BADDRCT	7-15	CORRECT THE DIRECTORY CARDS AND RELOAD THE CARD READER RESPOND WITH : GENERATE DIRECT
-ASMERR	7-15	CORRECT THE {DMKRIO|DMKSYS|DMKFCB|DMKSNT} ASSEMBLE FILE AND RELOAD THE CARD READER RESPOND WITH : GENERATE {DMKRIO|DMKSYS|DMKFCB|DMKSNT}
-CP	7-15	VIRTUAL=REAL OPTION REQUIRED (YES,NO):
	7-15	STORAGE SIZE OF VIRTᴸREAL <MINIMUM IS 32K>:
	7-15	** SIZE ROUNDED TO NEXT HIGHER 4K BOUNDARY **
	7-15	STORAGE SIZE FOR VIRTUALᴸREAL nnnnK
	7-15	IS THE ABOVE ENTRY CORRECT (YES,NO):
	7-15	ERROR WHILE WRITING "DMKSLC TEXT" FILE

Figure 7-11. GENERATE Messages (Part 2 of 3)

Label	Diagram	Message Text
-BUILD	7-15	IPLABLE NUCLEUS NOW ON TAPE **** WHEN 'NUCLEUS LOADED ON xxxxxx' IS TYPED, ISSUE 'CLOSE PRT', TO GET THE CPLOAD MAP. WHEN PRINTING IS COMPLETE, SHUTDOWN THE SYSTEM AND IPL THE NEW SYSRES VOLUME.
	7-15	TAPE (182) -- NOT READY OR NOT ATTACHED HIT RETURN WHEN READY OR 'EXIT':
	7-15	ERROR BUILDING xxxxxxxx NUCLEUS
	7-15	ERROR WRITING CP NUCLEUS TO TAPE
-REALIPL	7-15	TO LOAD THE CP NUCLEUS JUST CREATED, SHUTDOWN THE SYSTEM AND THEN IPL THE TAPE. THE CPLOAD MAP WILL AUTOMATICALLY BE PRINTED AT THE PRINTER WHOSE ADDRESS IS '00E'. IF THERE IS NO PRINTER AT THIS ADDRESS THE LOAD MAP WILL BE PRINTED AT THE FIRST PRINTER CAUSING AN INTERRUPT, (IE. NOT-READY TO READY SEQUENCE). ONCE THE NUCLEUS HAS BEEN LOADED, YOU MAY IPL YOUR NEW CP SYSTEM RESIDENCE VOLUME. NOTE: THERE MUST BE ENOUGH STORAGE ON THE SYSTEM (VIRTUAL OR REAL), TO CONTAIN THE VIRT=REAL AREA AND THE CP NUCLEUS.

Figure 7-11. GENERATE Messages (Part 3 of 3)

DMKLDOOE (LOADER) PROGRAM

If the loader terminates, one of the following wait conditions is indicated in the instruction counter:

Code	Meaning
X'111111'	A program check occurred.
X'222222'	A unit check occurred while the bootstrap routine was reading in the loader.
X'999999'	An SVC was issued.
X'BBBBBB'	A machine check occurred.
X'CCCCCC'	An I/O error occurred on the card reader.
X'FFFFFF'	An I/O error occurred for the console (X'00' contains the message UNRECOVERABLE ERROR), or the control card for changing the default I/O addresses for the printer or terminal is invalid (X'00' contains the message BAD DEVICE CARD or INVALID DEVICE SPECIFIED).

LOADER WAIT STATE CODES

If the instruction counter contains X'999999', indicating an SVC wait state, examine the interruption code (the third and fourth bytes of the supervisor old PSW). The interruption codes (shown in hexadecimal) have the following meanings:

Code	Meaning
64	An error occurred during conversion of a value from hexadecimal to binary format.
65	There is no more free storage available for the loader.
66	A duplicate type 1 ESD (External Symbol Dictionary) entry has been encountered.
67	The "name" in the LDT (Loader Terminate) statement is undefined.
68	The control section named in the ICS (Include Control Section) statement was not found by end of file.
69	The loader attempted to add another entry to the reference table, which would have caused the table to overflow.
6A	The object modules being loaded are about to overlay the loader.
6B	The object modules being loaded are about to overlay an address between zero and 100.
6C	A permanent error occurred in the input device.
6D	The loader is trying to release storage that is not on a doubleword boundary.

For further explanations of these wait state conditions and the recommended operator action to correct them, see *VM/370 System Messages*.

Chapter 8. The VM/370 Starter System

Introduction

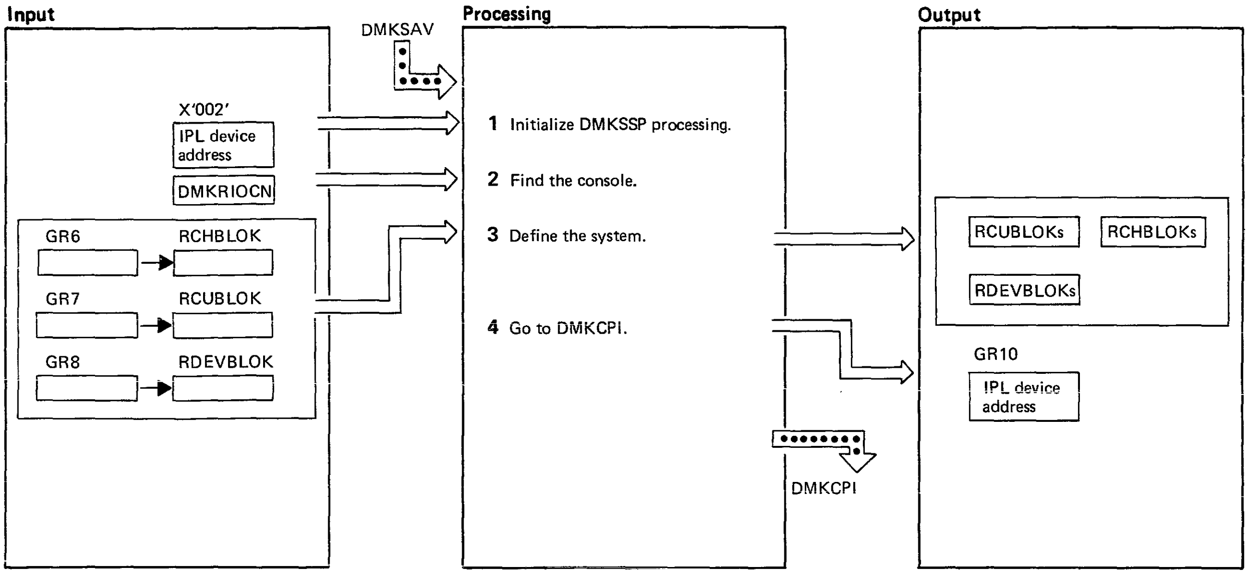
The Starter System Program (DMKSSP) redefines the real configuration according to the operator's specifications.
the DMKSAV module reads a copy of the

Normally, VM/370 is loaded from disk CP nucleus into real storage and then calls DMKCPI to perform the initialization tasks (such as initializing storage, mounting devices, and so on). However, during system generation, the VM/370 starter system is loaded from the starter system tape. When VM/370 is loaded from the starter system tape, the DMKSAV module reads a copy of the starter system nucleus into real storage and calls DMKSSP to give the operator the opportunity to redefine the devices necessary to continue with system generation. When DMKSSP is through with its processing, it calls DMKCPI to continue the initialization process.

DMKSSP is an interactive program. The operator must signal attention to define a console at an address other than 009 or 01F. Then, the operator responds to questions displayed at the terminal to redefine the printer, punch, reader, tape and disk devices.

Method of Operation

This section describes those functions that are performed by the DMKSSP program. There is only one method of operation diagram and that is Diagram 8-1.



Notes	Module	Label	Ref
1 Registers 11 and 12 are set up as base registers. The new I/O PSW, new machine check PSW, and new program check PSW are set up and all interrupts are disabled.	DMKSSP	DMKSSP01	
2 If the console address is valid, DMKSSP displays	DMKSSP		
VM/370 STARTER SYSTEM VERSION n.n		HDRMSG	
*** DO YOU WISH TO REDEFINE YOUR SYSTEM *** (YES, NO):		REDEFINE	
If the response is YES, proceed by redefining the system (see step 3). If the response is NO, DMKSSP processing is done. Proceed to step 4.			
3 First, all the control blocks and their pointers are cleared and the system residence device is set up.	DMKSSP	MAINLINE	
DMKSSP must find the console. If the console is not at 009 or 01F, DMKSSP enables for interrupts and waits until the operator signals attention to identify the console.		FINDCONS	
The CPU model is checked and if it is valid, DMKSSP builds the real control blocks for the console, and displays		VLDCON	
VM/370 STARTER SYSTEM VERSION n.n		HDRMSG	
DMKSSP prompts the operator to reconfigure the system. DMKSSP displays		PRTLAB	

ENTER PRINTER ADDRESS (cuu):
ENTER DEVICE TYPE (1403, 1443, 3211):			

and builds the printer real control blocks according to the operator's response.			
DMKSSP displays		PCHLAB	

ENTER DEVICE ADDRESS (cuu):
ENTER DEVICE TYPE (2540P, 3525):			

and builds the punch real control blocks according to the operator's response.			
and builds the reader real control blocks according to the operator's response.			
DMKSSP displays		PIDLAB	

ENTER ADDRESS WHERE PID TAPE IS MOUNTED (cuu):
ENTER DEVICE TYPE (2401, 2415, 2420, 3420):			

and builds the tape real control blocks according to the operator's response.			
DMKSSP displays		BKUPLAB	

ENTER ADDRESS WHERE SCRATCH TAPE IS MOUNTED (cuu):
ENTER DEVICE TYPE (2401, 2415, 2420, 3420):			

and builds the tape real control blocks according to the operator's response.			
DMKSSP displays		SYSLAB	

ENTER DEVICE ADDRESS WHERE SYSTEM RESIDENCE WILL BE BUILT (cuu):
ENTER DEVICE TYPE (2319, 2314, 3330, 3340, 2305):			

and builds the disk real control blocks according to the operator's response.			
DMKSSP then asks the operator to verify the configuration by displaying		WORKLAB	
*** SYSTEM DEFINITION COMPLETED *** cuu PRINTER cuu PUNCH cuu READER cuu PID TAPE cuu SCRATCH TAPE cuu NEW SYSTEM RESIDENCE cuu SCRATCH PACK ARE THE ABOVE ENTRIES CORRECT (YES,NO):			
If the operator responds NO, the entire system definition process is repeated.			
4 Control is transferred to DMKCPI with the address of the IPL device in general register 10.	DMKSSP	XPRINIT	

Diagram 8-1. DMKSSP--The Starter System

Program Organization

This section describes the organization of the DMKSSP module.

DMKSSP

The Starter System Program that allows the operator to redefine the minimum devices necessary to generate the VM/370 system.

Attributes
Nonreentrant, resident, entered via IPL.

Entry Conditions
DMKSSP001 is entered as the result of an IPL.

Exit Conditions
DMKSSP gives control to DMKCPINT to initialize the remainder of the system. Register 10 must contain the IPL device address.

Register Usage
R1: Parameter register
R2: Parameter register
R5: General BAL register
R6: Address of RCHBLOK
R7: Address of RCUBLOK
R8: Address of RDEVBLOK
R11: Base register 2
R12: Base register 1

External References
DMKRIODV Anchor to the first real device block
DMKRIOCU Anchor to the first real control unit block
DMKRIOCH Anchor to the first real channel block
DMKRIOCN Address of the system console device
DMKRIOPR Address of the system printer device
DMKRIOPU Address of the system punch device
DMKRIORD Address of the system reader device
DMKSYSNU Disk address on the nucleus
AMKRIO Address of real I/O control blocks

Call to Other Routines
DMKCVTHB To convert the device address to binary
DMKCVTBH To convert the device address to printable hexadecimal characters
DMKCPINT To continue system initialization

Data Areas
RCHBLOK, RCUBLOK, RDEVBLOK, PSA

Directory

Figure 8-1 is an alphabetic list of the major labels in the Starter System Program. The associated method of operation diagram (if any) is indicated and a brief description of the operation performed at the point in the program associated with each label is included.

Label	Diagram	Description
ATTNHAND	8-1	Enables system for I/O interrupts.
BKUPLAB	8-1	Builds real control blocks for scratch tape.
DASDADR	8-1	Sets up device type for disk containing the starter system.
DMKSSP01	8-1	Starter system entry point called by DMKSAV.
FINDCONS	8-1	Identifies the system console.
GRAPHID	8-1	Handles the I/O for display terminals.
HDRMSG	8-1	Displays starter system header message.
MAINLINE	8-1	Builds all the real control blocks necessary.
PCHLAB	8-1	Builds the real control blocks for the punch.
PIDLAB	8-1	Builds the real control blocks for the tape drive containing the PID (Program Information Department) distribution tape.
PRTLAB	8-1	Builds the real control blocks for the printer.
RDRLAB	8-1	Builds the real control blocks for the reader.
READADDR	8-1	Initiates writes to and reads from the console to determine the device address.
READTYPE	8-1	Initiates writes to and reads from the console to determine the device type.
REAWRITE	8-1	Writes to and reads from the console. The REAWRITE routine is called by both the READADDR and READTYPE routines.
REDEFINE	8-1	Asks the operator if he wants to redefine the system.
SCAN	8-1	Finds or builds the necessary real control blocks.
STARTIO	8-1	Issues the Start I/O (SIO).
SYSLAB	8-1	Builds the real control blocks for the disk that contains the system residence volume.
VLDCON	8-1	Checks for a valid CPU model.
WORKLAB	8-1	Asks the operator if the configuration just defined is the one he wants.
XFRINIT	8-1	Transfers control to DMKCPI.

Figure 8-1. The Starter System (DMKSSP) Label Directory

Diagnostic Aids

Figure 8-2 lists the messages issued by the Starter System Program. The associated program label and method of operation diagram are included in the list.

Label	Diagram	Message Text

BKUPLAB	8-1	ENTER ADDRESS WHERE SCRATCH TAPE IS MOUNTED (cuu):
		ENTER DEVICE TYPE (2401, 2415, 2420, 3420):

HDRMSG	8-1	VM/370 STARTER SYSTEM VERSION n.n

PCHLAB	8-1	ENTER PUNCH ADDRESS (cuu):
		ENTER DEVICE TYPE (2540P, 3525):

PIDLAB	8-1	ENTER ADDRESS WHERE PID TAPE IS MOUNTED (cuu):
		ENTER DEVICE TYPE (2401, 2415, 2420, 3420):

PRTLAB	8-1	ENTER PRINTER ADDRESS (cuu):
		ENTER DEVICE TYPE (1403, 1443, 3203, 3211, 3800):

RDRLAB	8-1	ENTER READER ADDRESS (cuu):
		ENTER DEVICE TYPE (2501, 2540R, 3505):

REDEFINE	8-1	***DO YOU WISH TO RE-DEFINE YOUR SYSTEM*** (YES,NO):

SYSLAB	8-1	ENTER DEVICE ADDRESS WHERE SYSTEM RESIDENCE WILL BE BUILT (cuu):
		ENTER DEVICE TYPE (2319, 2314, 3330, 3340, 3350, 2305):

WORKLAB	8-1	***SYSTEM DEFINITION COMPLETED*** cuu PRINTER cuu PUNCH cuu READER cuu PID TAPE cuu SCRATCH TAPE cuu NEW SYSTEM RESIDENCE cuu SCRATCH PACK ARE THE ABOVE ENTRIES CORRECT (YES,NO):
WNGDEV		**ERROR** DEVICE HAS BEEN ALREADY ALLOCATED

Figure 8-2. The Starter System (DMKSSP) Messages

Chapter 9. The 3704/3705 Service Programs

Introduction

There are four CMS commands and two CP commands specifically for generating and manipulating the 3704/3705 control program. The CMS commands are needed to generate and save a copy of the 3704/3705 control program. The CP commands allow you to operate and manipulate the 3704/3705 in a manner similar to the way other CP commands let you operate your other virtual machine devices.

The CMS commands that help you generate a 3704/3705 control program are: ASM3705, GEN3705, LKED, and SAVENCP. The ASM3705 command is an interface between CMS and the NCP/VS Release 2 and 3 Assembler (IFKASM) or the NCP/VS Release 4 Assembler (CWAX00). It accepts source statement files as input, checks that the input file exists and that the options specified are valid, calls IFKASM or CWAX00 to perform the assembly, and produces an object deck and program listing as output. The ASM3705 command produces the stage 1 output for the 3704/3705 control program generation process.

The GEN3705 command accepts the file produced in stage 1, creates a unique assembler file for each job step in the input file, creates several unique files containing the linkage editor statements necessary to build the load module file, and builds an EXEC macro file of the CMS commands necessary to assemble and load the 3704/3705 control program. If SAVE was specified on the command line, it saves a copy of the control program in page-format on a CP-owned volume.

The LKED command is an interface between CMS and the OS/VS1 linkage editor. The GEN3705 command processor embeds the LKED commands in the EXEC macro file it produces. The LKED command processor interprets the CMS command lines, defines the necessary files, and links to the OS/VS linkage editor. Two permanent files are produced: the 'filename LOADLIB' file, which contains the load modules, and the 'filename LKEDIT' file, which contains the printed output.

The SAVENCP command builds the parameter list (CCPARM) and calls DMKSNC via Diagnose instruction X'50' to write a core image copy of the 3704/3705 control program to a CP-owned system volume. This copy of the control program is loaded each time the 3704/3705 is loaded.

The CP commands that help you to control the operation of the 3704/3705 are NCPDUMP and NETWORK. The NCPDUMP command processor performs several different tasks. It:

- Erases a specific CP or CMS 3704/3705 dump file.
- Formats the 3704/3705 dump.
- Prints the 3704/3705 dump file.
- Assigns an identifier to the 3704/3705 dump file.
- Creates the CMS 3704/3705 dump file.

The NETWORK command processor provides the support for the 3704/3705 that several CP commands (ENABLE, DISABLE, QUERY, DISPLAY, VARY, HALT, TRACE, and SHUTDOWN) provide for other devices. In addition, the NETWORK command has options that load a named 3704/3705 control program into 3704/3705 storage and dump the contents of that storage.

These commands are discussed in detail in other publications. For more information about the ASM3705, GEN3705, LKED, and SAVENCP commands and a complete description of the generation process, see the VM/370 Planning and System Generation Guide. For more information about the NCPDUMP and NETWORK commands, see the VM/370 Operator's Guide.

The ZAP service program, which allows you to update and dump existing 3704/3705 load libraries, is described in "Chapter 10. The ZAP Service Program" and in the VM/370 Operator's Guide.

Method of Operation

This section describes the CMS modules that provide the commands to generate the 3704/3705 control programs. Diagrams describe the functions performed by each of the command processors. Figure 9-1 shows the relationships between these diagrams.

Diagram 9-1 describes the SAVENCP command, which saves an image of the 3704/3705 control program so that it can later be loaded. Diagram 9-2 shows how CCPARM is built.

Diagrams 9-3, 9-4, and 9-5 describe the GEN3705 command, which generates a series of commands to assemble, link edit, and load the 3704/3705 control program.

Diagrams 9-6 and 9-7 describe the ASM3705 command, which is an interface between CMS and the NCP/VS Assembler (IFKASM or CWAX00).

Diagram 9-8 describes the LKED command, which is an interface between CMS and the OS/VS1 Linkage Editor.

Diagram 9-9 describes the NCPDUMP command, which prints a dump of the 3704/3705 storage.

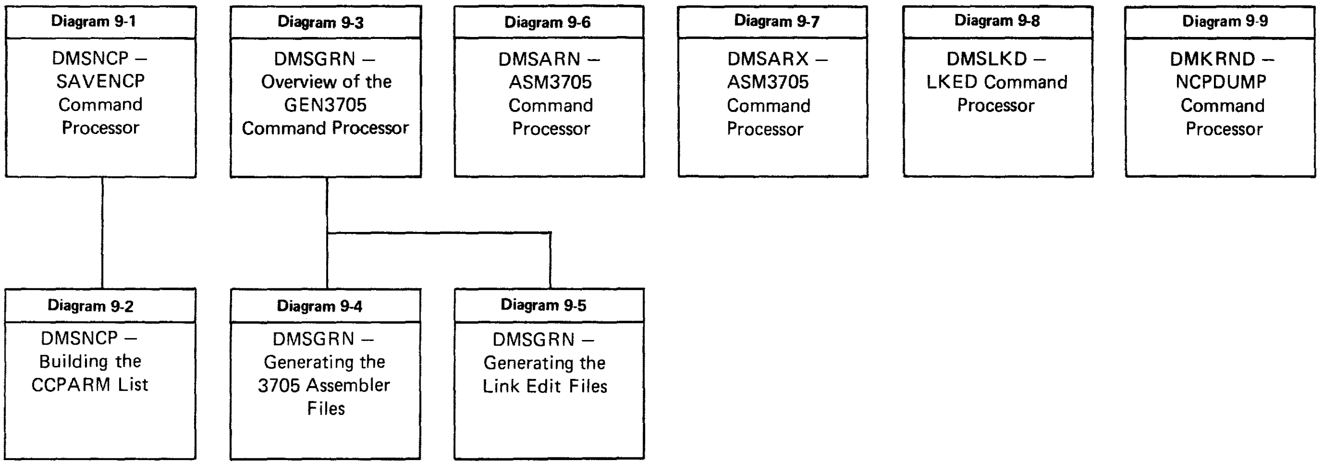
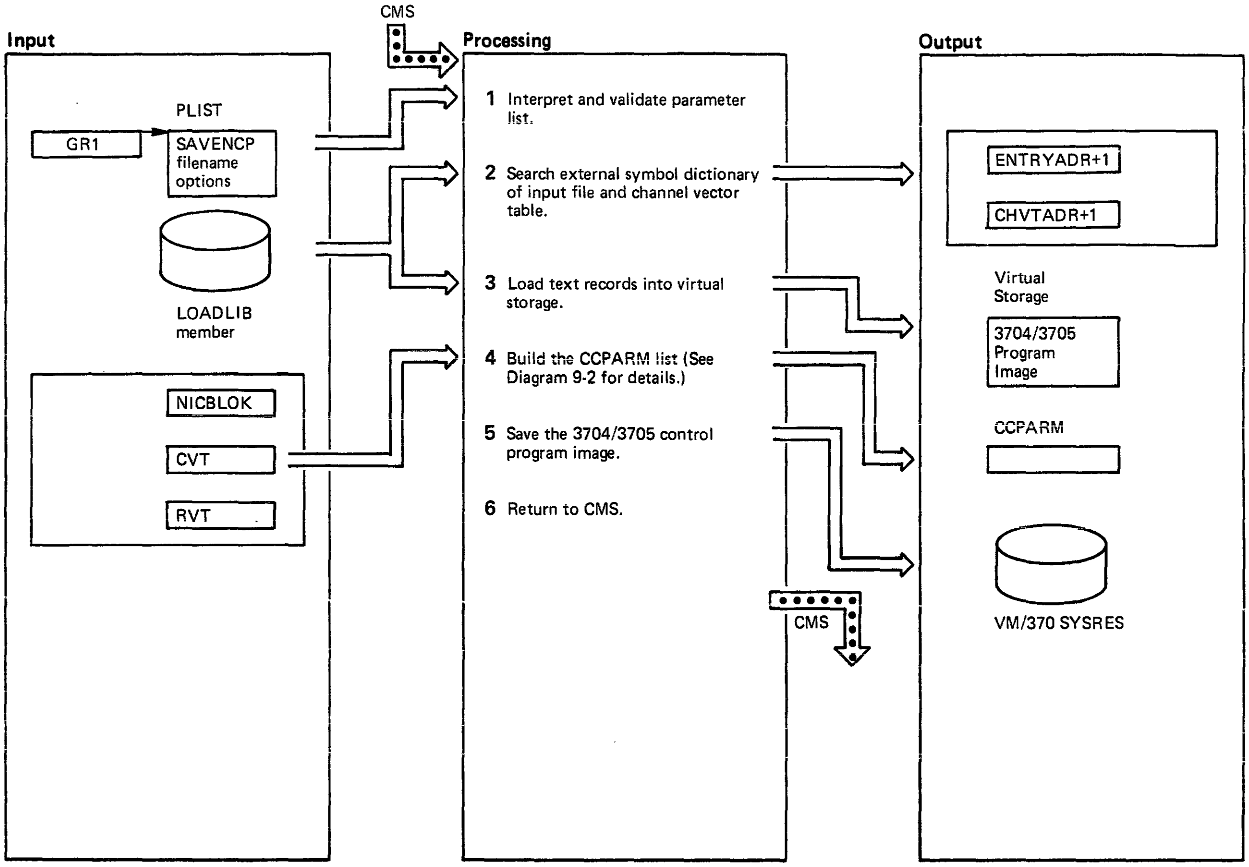


Diagram 9-1
DMSNCP – SAVENCP Command Processor
Diagram 9-3
DMSGRN – Overview of the GEN3705 Command Processor
Diagram 9-6
DMSARN – ASM3705 Command Processor
Diagram 9-7
DMSARX – ASM3705 Command Processor
Diagram 9-8
DMSLKD – LKED Command Processor
Diagram 9-9
DMKRND – NCPDUMP Command Processor
Diagram 9-2
DMSNCP – Building the CCPARM List
Diagram 9-4
DMSGRN – Generating the 3705 Assembler Files
Diagram 9-5
DMSGRN – Generating the Link Edit Files

Figure 9-1. Key to the 3704/3705 Service Programs Method of Operation Diagrams



1 Interpret and validate parameter list.

2 Search external symbol dictionary of input file and channel vector table.

3 Load text records into virtual storage.

4 Build the CCPARM list (See Diagram 9-2 for details.)

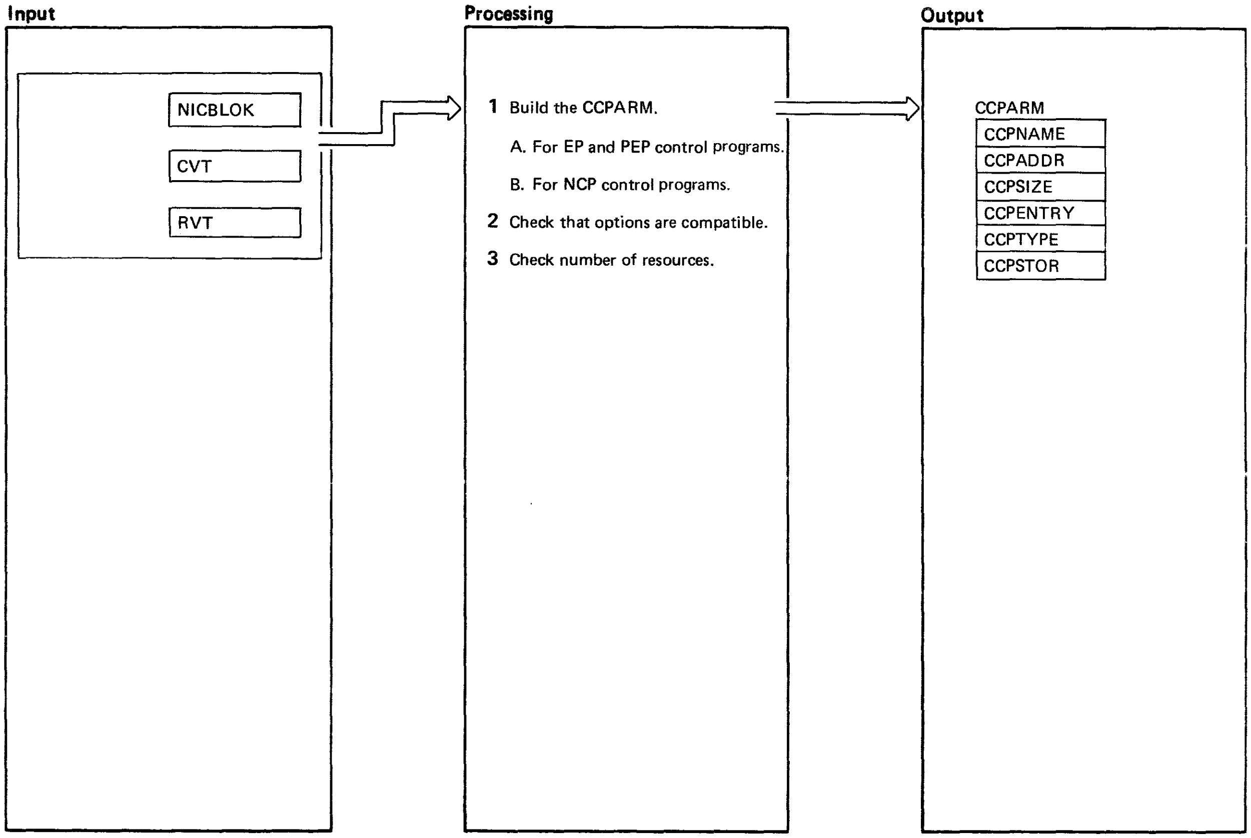
5 Save the 3704/3705 control program image.

6 Return to CMS.

Notes	Module	Label	Ref
1 The filename must be specified. If a library name or a member name is not specified, the input filename is used. If the 3704/3705 control program load module entry point is not specified, CXFINIT is assumed.	DMSNCP	SAVENCP	
An error in the parameter list results in one of the following messages DMSNCP001E NO FILENAME SPECIFIED DMSNCP002E FILE 'fn ft fm' NOT FOUND DMSNCP003E INVALID OPTION 'option' being issued and control being returned to CMS with return code 24 or 28. If no errors are encountered, the input file is opened and a search is made for the member. When the member is found, it is read. If the member is not found, the message DMSNCP013E MEMBER xxxxxxxx NOT FOUND IN LIBRARY is issued and control returns to CMS with a return code of 4.		ENDPARMS DOSTATE	
2 The entry point for NCP or PEP is CXFINIT. The entry point for EP is CYASTART. For either EP or PEP, the channel vector table, CYACHVT, CYECHVT1, or CYECHVT2 must also be found. The entry point address and channel vector table address are saved.	DMSNCP	CESDENT CESDCHVT	

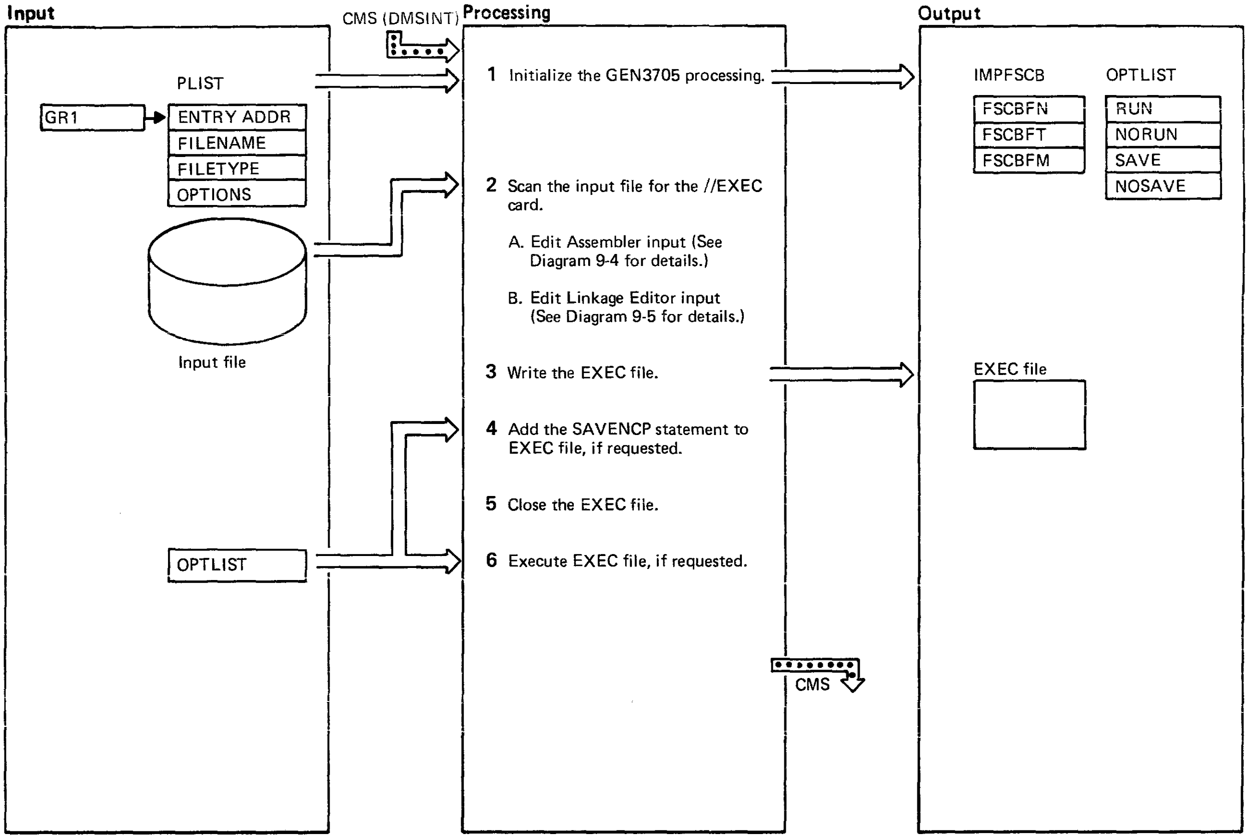
Notes	Module	Label	Ref
3 The text records are moved from the input buffer into the proper position in the core image buffer. If the entry point symbol has not been resolved when the first text record is encountered, the message	DMSNCP	CONTROL	
DMSNCP021E ENTRY POINT xxxxxxxx NOT FOUND is issued and control returns to CMS with a return code of 40. Premature end of file or invalid control records cause the messages		ERR21	
DMSNCP056E FILE 'fn ft' CONTAINS INVALID RECORD FORMATS DMSNCP109E VIRTUAL STORAGE CAPACITY EXCEEDED to be issued and control to be returned to CMS.		ERR66	
4 When the core image buffer is loaded, the input file is closed. The Communication Control Parameter list (CCPARM) is built from the information in the core image buffer.	DMSNCP	CLOSE	
5 The size of the read buffer is stored in register 1 and the DIAGNOSE instruction with code X'50' is issued to save a copy of the 3704/3705 control program.	DMSNCP	SAVECCP	
6 The return code from the DIAGNOSE instruction is passed to CMS and control returns to CMS.	DMSNCP	EXIT	

Diagram 9-1. DMSNCP--SAVENCP Command Processor


Input
NICBLOK
CVT
RVT
Processing
1 Build the CCPARM.
A. For EP and PEP control programs.
B. For NCP control programs.
2 Check that options are compatible.
3 Check number of resources.
Output
CCPARM
CCPNAME
CCPADDR
CCPSIZE
CCPENTRY
CCPTYPE
CCPSTOR

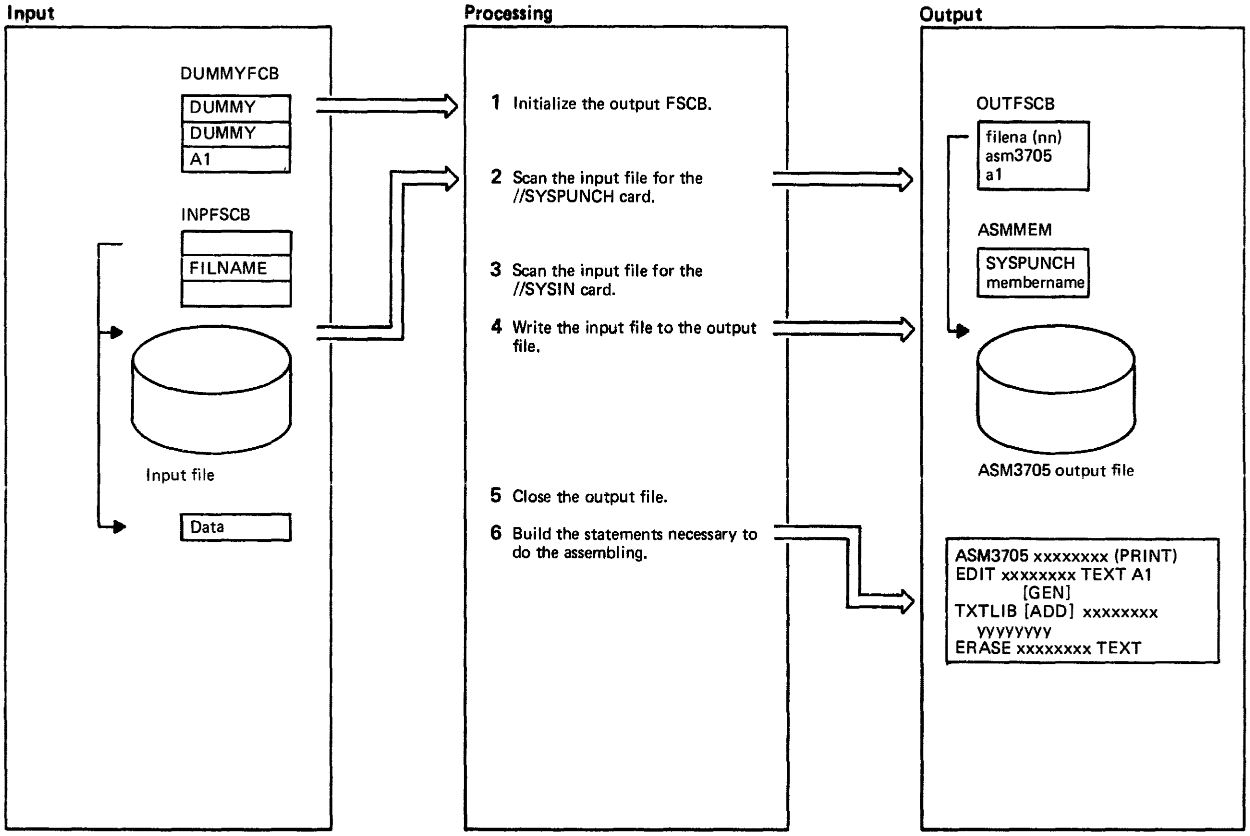
Notes	Module	Label	Ref
1	DMSNCP		
A. For EP and PEP control programs, additional fields are updated (CCPRSTYP, CCPRSTAT, CCPRSTEP, CCPPSIZE). A channel vector table must exist for EP and PEP control programs. If the CVT does not exist, the message DMSNCP025E INVALID DATA IN 370X PROGRAM is issued and control returns to CMS with return code 16.		SCANCEP	
B. Additional fields in the CCPARM block are updated for NCP and PEP control programs (CCPCAONE, CCPHBFSZ, CCPHBFNO, CCPPADO, CCPPADI, CCPMAXID, CCPRESID, CCPRSTYP, CCPRSTAT CCPRSTEP).		SCANNCP	
2 A check is made that the options specified are compatible. If they are not, the message DMSNCP099W GENERATION PARAMETERS INCOMPATIBLE WITH VM/370 is issued and processing continues.	DMSNCP	CHEKVMV	
3 If there are more than 4086 resources or if the first resource is not a 3704/3705, the message	DMSNCP		
DMSNCP025E INVALID DATA IN 370X PROGRAM is issued and control returns to CMS with a return code of 16.			

Diagram 9-2. DMSNCP--Building the CCPARM List


Input
PLIST
GR1
ENTRY ADDR
FILENAME
FILETYPE
OPTIONS
Input file
OPTLIST
CMS (DMSINT)
Processing
1 Initialize the GEN3705 processing.
2 Scan the input file for the //EXEC card.
A. Edit Assembler input (See Diagram 9-4 for details.)
B. Edit Linkage Editor input (See Diagram 9-5 for details.)
3 Write the EXEC file.
4 Add the SAVENCP statement to EXEC file, if requested.
5 Close the EXEC file.
6 Execute EXEC file, if requested.
CMS
Output
IMPFSCB
FSCBFN
FSCBFT
FSCBFM
OPTLIST
RUN
NORUN
SAVE
NOSAVE
EXEC file

Notes	Module	Label	Ref
1 The input file name, type, and optionally the mode are put into INPFSCB. The filename or the first 6 characters of the name, whichever is the least, is saved for naming the assembler and linkage editor output files.	DMSGRN	START	
The input options are scanned and the appropriate options are set on. Invalid options cause the message DMSGRN003E INVALID OPTION xxxxxxxx to be issued.		OPTIONS1	
The FSSTATE macro is issued to see if the file exists. Either of the following messages is issued in case of an error DMSGRN048E INVALID MODE xxx DMSGRN002E FILE xxxxxxxx NOT FOUND		OPTEND	
2 The FSCBRD routine is used to read the input file. The EDITIN routine scans for a //EXEC card containing PGM=IFKASM or PGM=IEWL. Control cards are scanned until a valid EXEC card is found. If *, //, or /* does not appear as the first characters of the input record or if an invalid //EXEC card is read, the message DMSGRN078E INVALID CARD IN INPUT FILE 'xxxxxxxxxxxxxxx' is displayed.	DMSGRN	PRIMEDIT	
The IFKASM routine processes the assembler input and the IEWL routine processes the linkage editor input. After the input is processed, DMSGRN continues by scanning the input file for another //EXEC card.		FINDASM FINDIEWL	
3 The EXEC statements that were generated as a result of the assembler and linkage editor input are written to an EXEC file.	DMSGRN	STACK30	
4 The CLOSTACK routine is called to add SAVECP filename (ENTRY entryname to the end of the EXEC file, if SAVE was specified on the GEN3705 command.	DMSGRN	PROCEND2 STACK30	
5 The EXEC macro file is closed by branching and linking to the PROCEND routine.	DMSGRN	PROCEND1	
6 If RUN was specified, the command EXEC ncpname is stacked in the reader.	DMSGRN	PROCEND1	
Control is returned to CMS.		RETURN1	

Diagram 9-3. DMSGRN--Overview of the GEN3705 Command Processor



1 Initialize the output FSCB.

2 Scan the input file for the //SYSPUNCH card.

3 Scan the input file for the //SYSIN card.

4 Write the input file to the output file.

5 Close the output file.

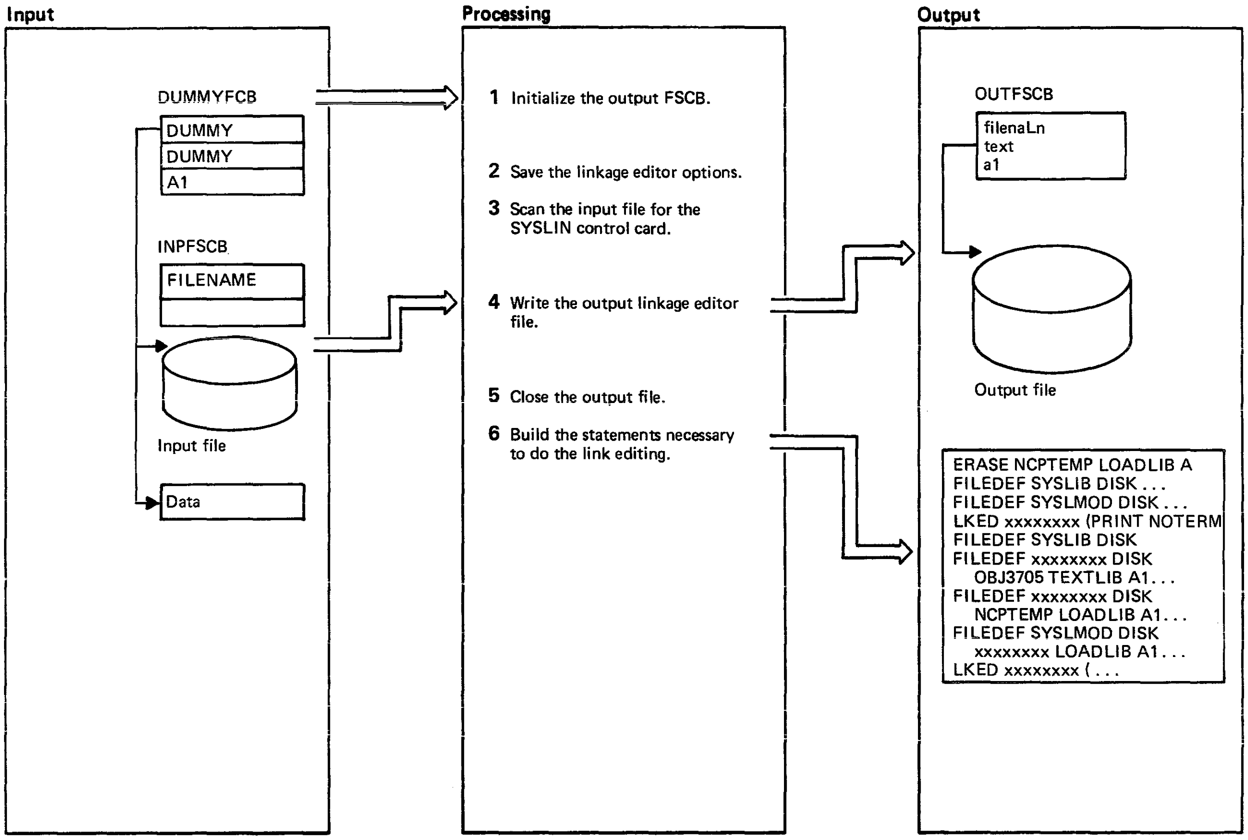
6 Build the statements necessary to do the assembling.

```
ASM3705 xxxxxxxx (PRINT)
EDIT xxxxxxxx TEXT A1
        [GEN]
TXTLIB [ADD] xxxxxxxx
   yyyyyyyy
ERASE xxxxxxxx TEXT
```

Notes	Module	Label	Ref
1 The filetype in the dummy FSCB is initialized to ASM3705. Each ASM3705 file has a filename consisting of the first 6 characters of the filename (or the entire filename if it is 6 characters or less) concatenated with a number. The FSCBWT routine uses the dummy FSCB to initialize the OUTFSCB.	DMSGRN	IFKASM	
2 The input file is scanned for a SYSPUNCH or SYSPUNCH continuation card. If found, it is scanned for the DSN= or DSNAME= keyword.	DMSGRN	IFKASM10	
The DSNEDIT routine then saves the membername of the data set in the current SYSPUNCH membername savearea.		IFKASM34	
3 The input file is scanned for the SYSIN card. All cards scanned preceding the SYSIN card must have * or // in the first positions of the card. Otherwise DMSGRN078E INVALID CARD IN INPUT FILE 'xxxxxxxxxxxxxxx' is issued.	DMSGRN	IFKASM40	
4 The FSCBRD routine reads all the input and the FSCBWT routine writes it to the output file.	DMSGRN	IFKASMA0	
5 The output file is closed by branching and linking to the FSCBCLOS routine. Close errors are ignored.	DMSGRN	IFKASMK0	

Notes	Module	Label	Ref
6 The ASMFIRST bit in the PROCSW1 byte is tested. If the bit is on, the GEN parameter in the TXTLIB command is changed to ADD. Otherwise, the bit is turned on.	DMSGRN	ASMSTAK	
The name of the output assembler file is moved into the ASM3705 and EDIT commands. The FSCB base address is changed and the name of the input file is put into the TXTLIB command.		ASMSTAK2	
The SYSPUNCH membername is then moved to the TXTLIB command.		ASMSTAK4	
The number of commands and the address of the first command in the stack are loaded from STACKASM into registers 1 and 2 respectively.		ASMSTAK6	

Diagram 9-4. DMSGRN--Generating the 3705 Assembler Files



1 Initialize the output FSCB.

2 Save the linkage editor options.

3 Scan the input file for the SYSLIN control card.

4 Write the output linkage editor file.

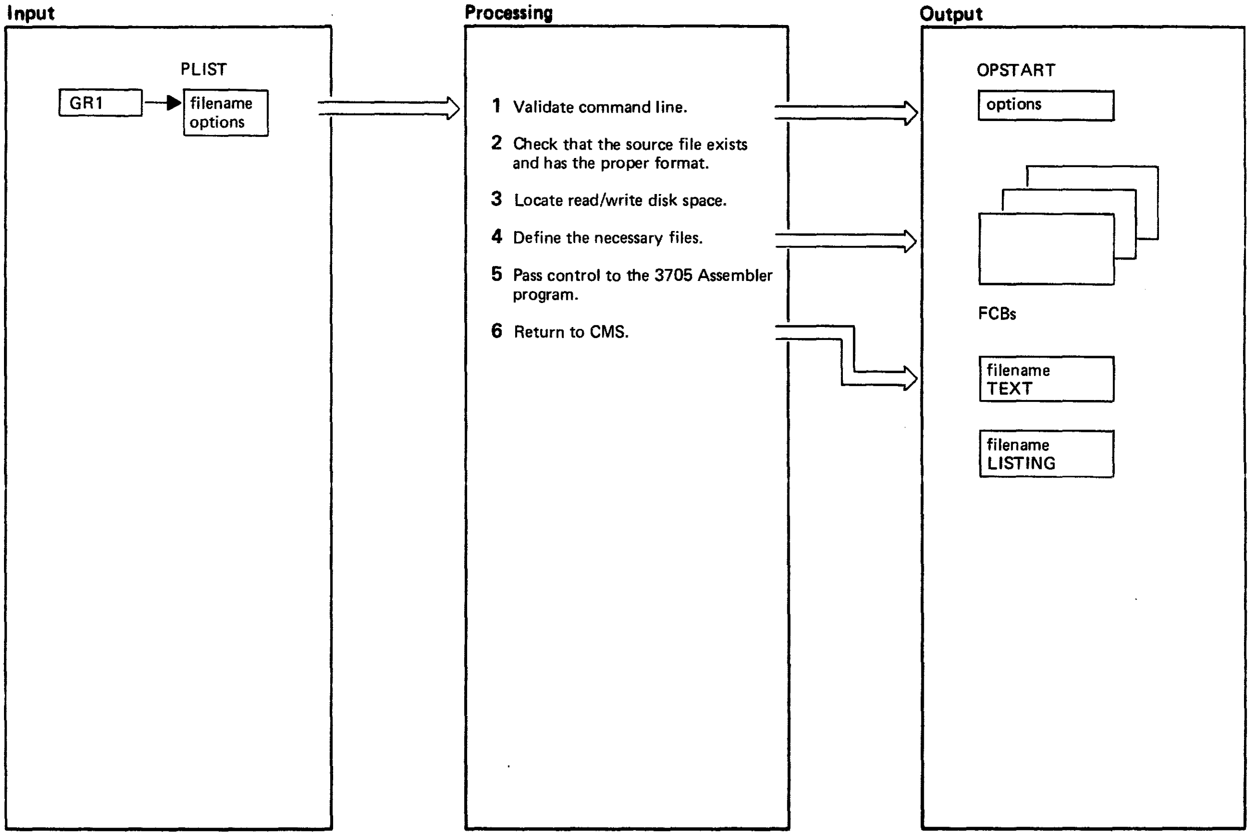
5 Close the output file.

6 Build the statements necessary to do the link editing.

```
ERASE NCPTEMP LOADLIB A
FILEDEF SYSLIB DISK . . .
FILEDEF SYSLMOD DISK . . .
LKED xxxxxxxx (PRINT NOTERM
FILEDEF SYSLIB DISK
FILEDEF xxxxxxxx DISK
   OBJ3705 TEXTLIB A1. . .
FILEDEF xxxxxxxx DISK
   NCPTEMP LOADLIB A1. . .
FILEDEF SYSLMOD DISK
   xxxxxxxx LOADLIB A1. . .
LKED xxxxxxxx ( . . .
```

Notes	Module	Label	Ref
1 The filetype in the dummy FSCB is initialized to TEXT. Each linkage editor TEXT file has a filename consisting of the first 6 characters of the filename (or the entire filename if it is 6 characters or less) concatenated with L and a number.	DMSGRN	IEWL	
2 The //EXEC card is edited for the keyword PARM=. The linkage editor options are moved to the option field of the LKED command. EXEC continuation cards are ignored.	DMSGRN	IEWLJCLA	
3 The input file is scanned for the SYSLIN card. All cards scanned preceding the SYSLIN card must have * or // in the first positions. Otherwise, the error message DMSGRN078E INVALID CARD IN INPUT FILE 'xxxxxxxxxxxxxxx' is issued.	DMSGRN	IEWLJCL2	
4 The FSCBRD routine reads the input file and the FSCBWT routine writes it to the output file.	DMSGRN	IEWLSN10 WRTSIN	
The EDITIN routine scans for the keyword ENTRY. If the keyword ENTRY is found, the IEWLENT routine moves the entry name to the SAVENCP statement.		IEWLENT	
5 The output file is closed by branching and linking to the FSCBCLOS routine.	DMSGRN	IEWLSEOF	
Close errors are ignored.		FSCBCLOS	
6 The LKDFIRST bit in the PROCSW1 byte is tested. If it is off, it is set on and the filename of the input file is moved into the FILEDEF and LKED commands. Also, the command count and address from STAKLKD1 are loaded into registers 1 and 2.	DMSGRN	LKDSTACK	
If the LKDFIRST bit is on, the command count and address from STACKLKD2 are loaded into registers 1 and 2.		LKDSTAK1	

Diagram 9-5. DMSGRN--Generating the Link Edit Files



1 Validate command line.

2 Check that the source file exists and has the proper format.

3 Locate read/write disk space.

4 Define the necessary files.

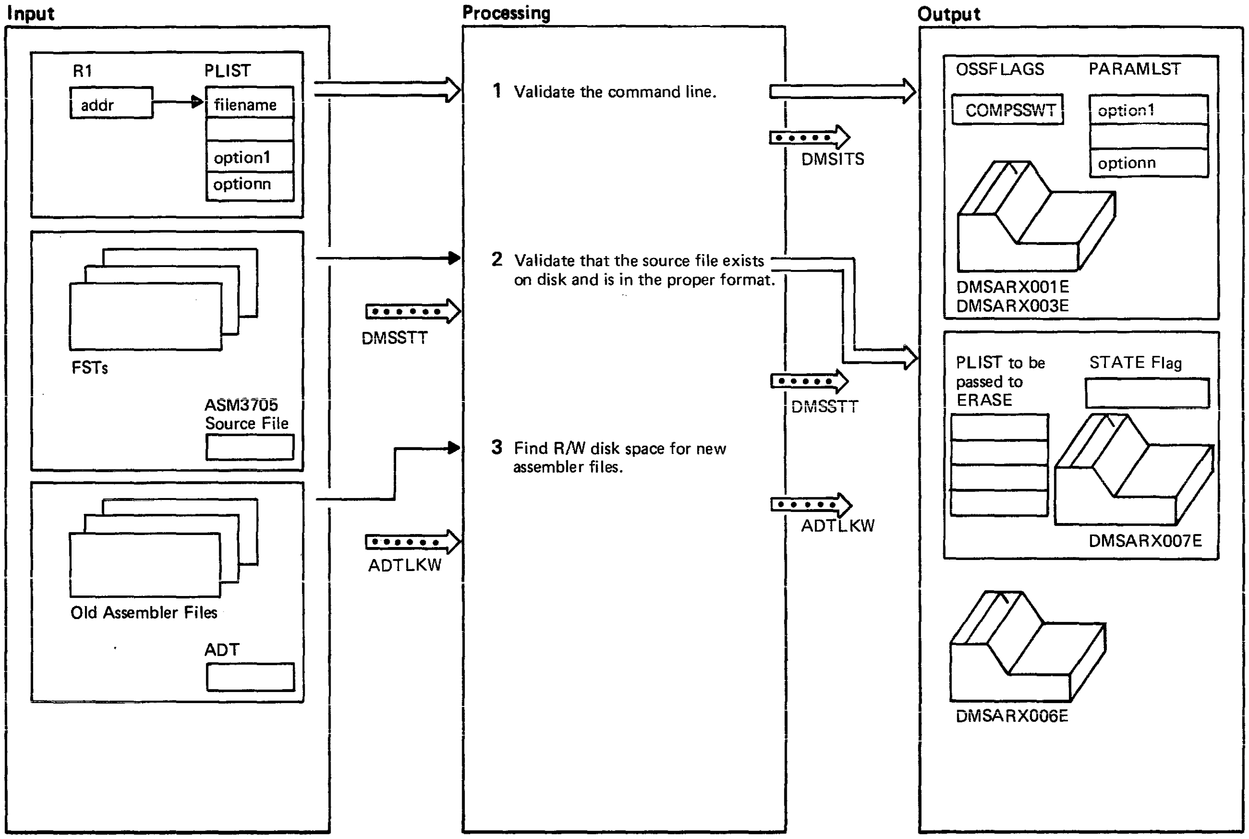
5 Pass control to the 3705 Assembler program.

6 Return to CMS.

Notes	Module	Label	Ref
1 A filename must be specified. If it is not, the message DMSARN001E NO FILENAME SPECIFIED is issued and processing terminates.	DMSARN	DMSARN	
The COMPSWT bit is set on in OSSFLAGS to indicate the 3705 assembler is running. The option list to be passed to the 3705 assembler is built.		SQUEEZE	
If Batch is running, the message ASSEMBLING filename A1 is displayed and steps 2 and 3 are skipped.		SUIT15	
2 The STATE macro is issued to check that the input file exists and has fixed 80-character records. If the record format is wrong, the message DMSARN007E FILE filename IS NOT FIXED, 80 CHAR. RECORDS is issued and processing terminates.	DMSARN	SUIT25	
3 If the input file resides on a read/write disk, that disk is used to contain the text and listing files that are generated. If the input disk is an extension of a read/write disk, the parent disk is used. Otherwise, the A disk is used.	DMSARN	SUIT17	

Notes	Module	Label	Ref
4 All the old text, listing, and utility files for the current file are erased. Free storage is initialized and enough storage to contain the longest assemble path is obtained via a GETMAIN call.	DMSARN	CONTINUE	
FILEDEFs are issued for SYSUT1, SYSUT2, SYSUT3, SYSIN, TEXT, SYSPUNCH (if the DECK option was specified), SYSPRINT (if the NOPRINT option was not specified), LISTING, and CMSLIB.		NOERASE	
5 Control is passed to IFKASM.	DMSARN	LIST2	
6 If the return code is not zero, one of the following messages is issued DMSARN004W WARNING MESSAGES ISSUED DMSARN008W ERROR MESSAGES ISSUED DMSARN012W SEVERE ERROR MESSAGES ISSUED DMSARN016W TERMINAL ERROR MESSAGES ISSUED	DMSARN	RETURN	
The output files are closed and the utility files SYSUT1, SYSUT2, and SYSUT3 are erased. All FCBs are cleared, OSSFLAGS is reset, and control returns to CMS.		SUIT19	

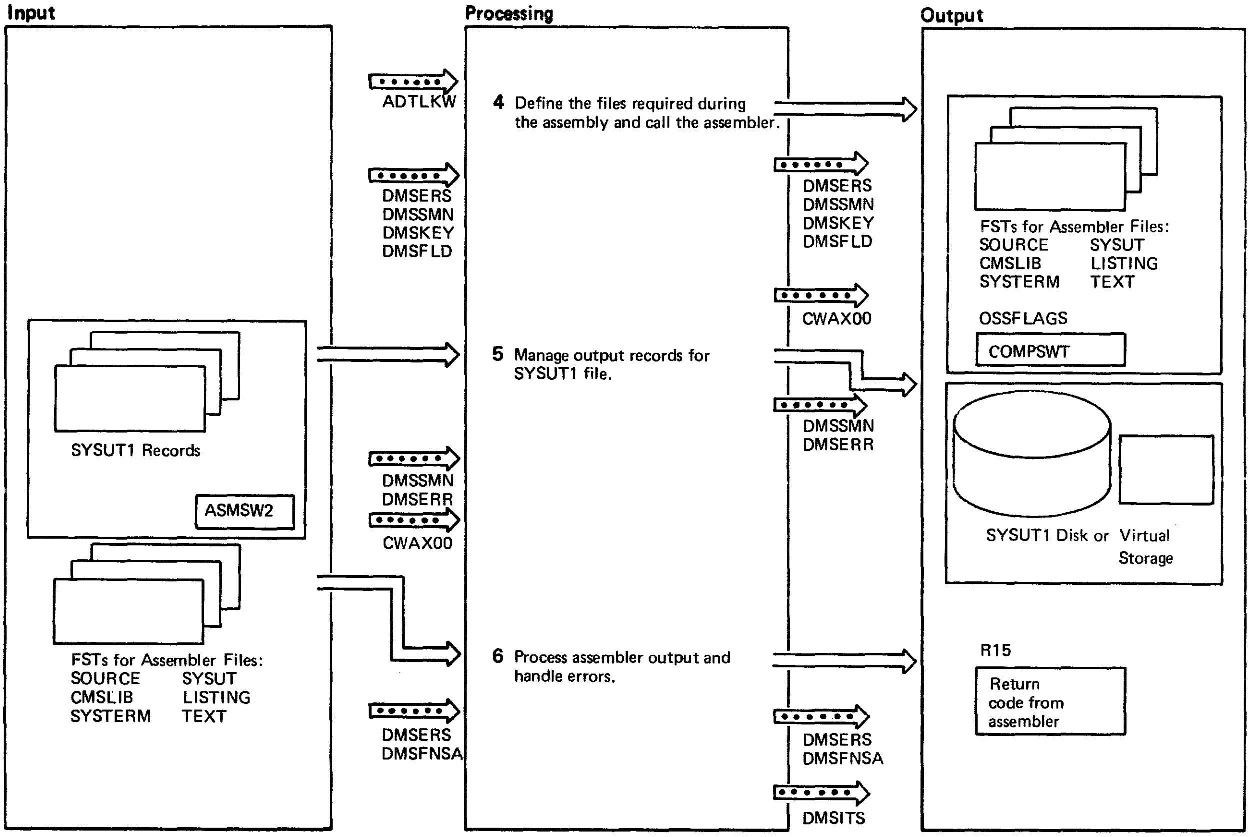
Diagram 9-6. DMSARN--ASM3705 Command Processor



Notes	Module	Label	Ref
1 Validate the command line by ensuring that a filename has been specified and creating an assembler option list. If the filename is not specified, the message DMSARX001E NO FILENAME SPECIFIED is issued. The option list is built by scanning the command line, checking the options specified, and placing the valid entries in the PARAMLST table. If an invalid option is specified, the message DMSARX003E INVALID OPTION 'option' is issued and processing terminates.	DMSARX	OPTSCN	
2 Verify that the source file exists by issuing a STATE command (module DMSSTT). If the file exists but is not in proper format (80-character records), the message DMSARX007E FILE 'fn ASM3705' IS NOT FIXED, 80-CHAR. RECORDS is issued and processing terminates. If the file is in proper format, processing continues at step 3.	DMSARX	STATASM	

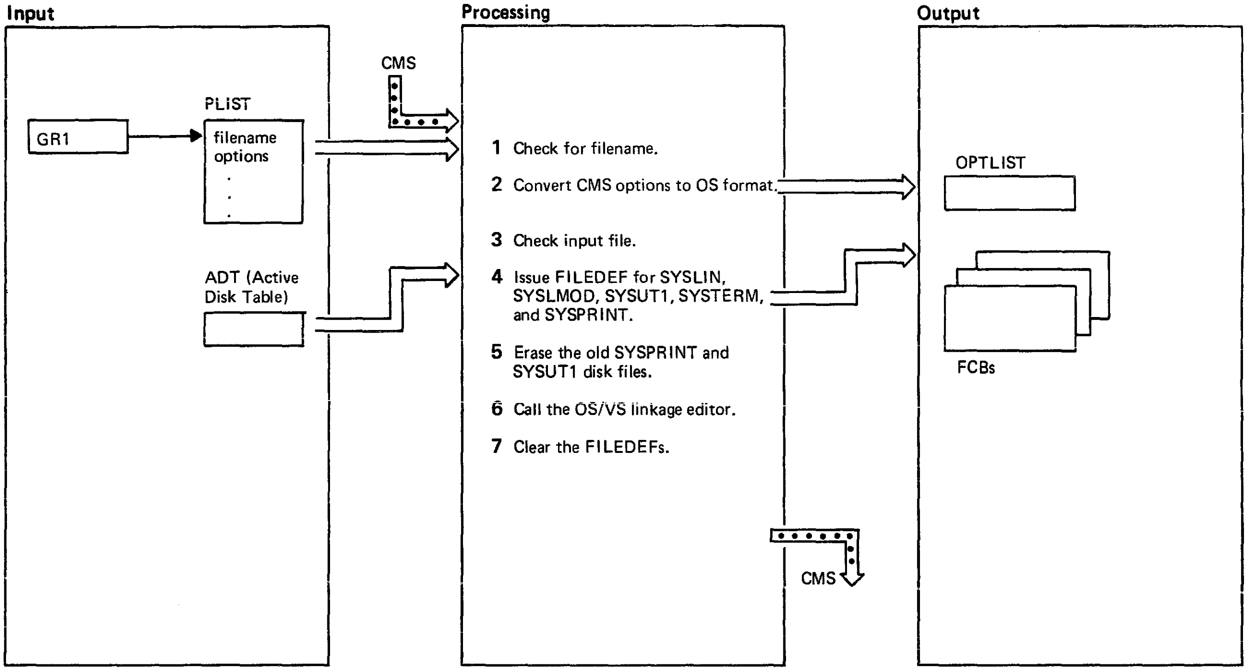
Notes	Module	Label	Ref
3 New files to be used during assembler processing (TEXT, LISTING, and SYSUT) can be obtained from three sources. If the input file resides on a R/W disk, that disk is used to contain the TEXT and LISTING files generated during the assembly. If the input file resides on an extension of the R/W disk, the parent disk is used. If neither of the above disks is a R/W disk, the user's A-disk is used. If no R/W disk can be obtained, the message DMSARX006E NO READ/WRITE DISK ACCESSED is issued and control returns to CMS via DMSITS.	DMSARX	FINDRW	

Diagram 9-7. DMSARX--ASM3705 Command Processor (Part 1 of 2)


Input
Processing
Output
ADTLKW
4 Define the files required during the assembly and call the assembler.
DMSERS
DMSSMN
DMSKEY
DMSFLD
DMSERS
DMSSMN
DMSKEY
DMSFLD
FSTs for Assembler Files:
SOURCE SYSUT
CMSLIB LISTING
SYSTERM TEXT
CWAX00
OSSFLAGS
COMPSWT
5 Manage output records for SYSUT1 file.
SYSUT1 Records
ASMSW2
DMSSMN
DMSERR
DMSSMN
DMSERR
CWAX00
SYSUT1 Disk or Virtual Storage
FSTs for Assembler Files:
SOURCE SYSUT
CMSLIB LISTING
SYSTERM TEXT
6 Process assembler output and handle errors.
R15
Return code from assembler
DMSERS
DMSFNSA
DMSERS
DMSFNSA
DMSITS

Notes	Module	Label	Ref
4 DMSERS is called to erase the old TEXT, LISTING, and SYSUT files associated with the new input file. DMSSMN (GETMAIN) is called to obtain enough storage to contain the SYSUT1 work file.	DMSARX	ERASE	
When disk space is obtained for the required assembler files and for the files CMS needs (SYSTERM and CMSLIB), FILEDEF commands are issued to convert all the files to CMS format. The assembler is then called and begins processing.		FILEDEF LOADASM	
5 If possible, all SYSUT1 records are kept in virtual storage during an assembly. However, when virtual storage is exhausted, records are written to disk.	DMSARX	ASMPROC	
If the records must be written to disk, they are formatted to fit DASD requirements and moved to disk a record at a time.		SYSWTX	
6 All SYSUT files used during the assembly are erased via a call to DMSERS. DMSFNSA is called to close all files and DMSFLD is called to clear all FILEDEFs not defined with the PERM option. COMPSWT in OSSFLAGS is turned off to indicate that the assembler is no longer processing, the auxiliary directory list is released, and control returns to CMS via DMSITS.	DMSARX	ERASUTS RETURN	

Diagram 9-7. DMSARX--ASM3705 Command Processor (Part 2 of 2)



1 Check for filename.

2 Convert CMS options to OS format.

3 Check input file.

4 Issue FILEDEF for SYSLIN, SYSLMOD, SYSUT1, SYSTERM, and SYSPRINT.

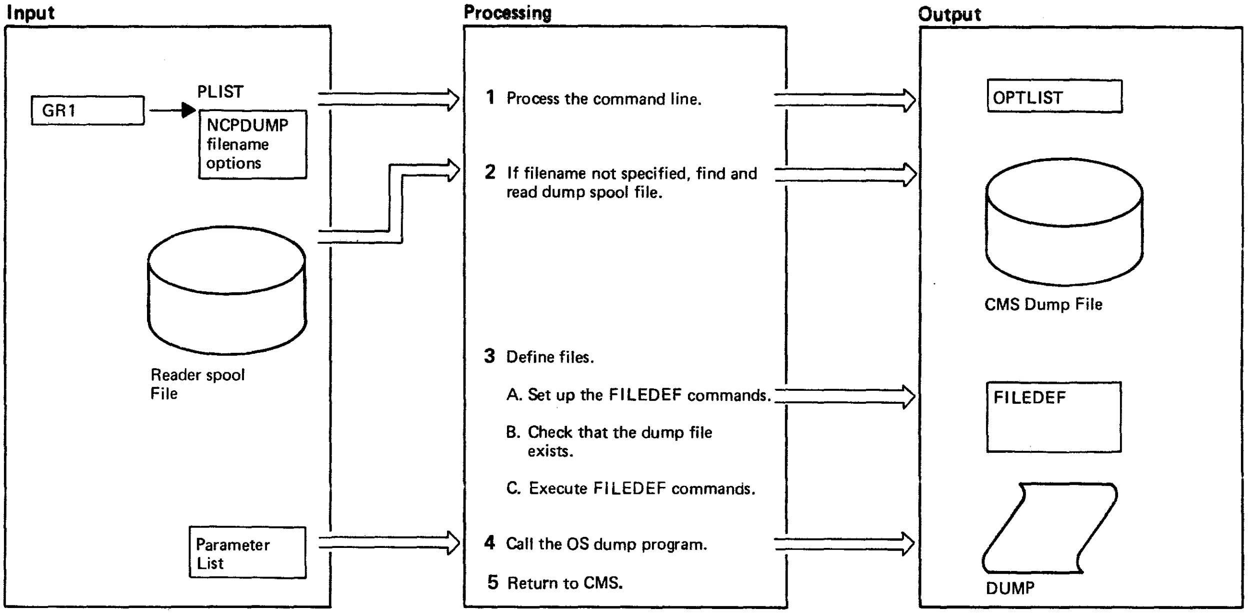
5 Erase the old SYSPRINT and SYSUT1 disk files.

6 Call the OS/VS linkage editor.

7 Clear the FILEDEFs.

Notes	Module	Label	Ref
1 The first operand on the LKED command must be the filename. If it is not, the message DMSLKD001E NO FILENAME SPECIFIED is displayed. The filename specified is used as the default FILEDEF filename.	DMSLKD	DMSLKD	
2 If anything other than options follows the filename, the message DMSLKD070E INVALID PARAMETER 'parameter' is issued. Flags are set to reflect the following options – PRINT, NOPRINT, DISK, SIZE, NAME, TERM, NOTERM, and LIBE. If they are specified, membername and libraryname are moved into the FILEDEF commands. If NAME or LIBE is specified without a corresponding name, the message DMSLKD005E NO 'option' SPECIFIED is issued.	DMSLKD	OUTLOOP2	
3 The STATE macro is issued to check that the input file exists. If it does not, the message DMSLKD002E FILE 'fn ft' NOT FOUND is issued. If the input file does not contain fixed 80-character records, the message DMSLKD007E FILE 'fn ft' IS NOT FIXED, 80 CHAR. RECORDS is issued.	DMSLKD	OUTLOOP2	
4 The CMS file definition function is called to create a file control block for each of the linkage editor DDNAMEs: SYSLIN, SYSLMOD, SYSUT1, SYSTERM, SYSPRINT. Standard file definitions are performed unless otherwise specified on the command line. If no read/write disk is accessed for the SYSUT1 file, the message DMSLKD006E NO READ/WRITE DISK ACCESSED is issued.	DMSLKD	PRTDEF	
5 The CMS erase function is called to delete 2 disk files: 'fn SYSUT1' and 'fn LKEDIT' (fn = the input filename).	DMSLKD	PRTDEF	
6 Control is passed to the OS/VS1 linkage editor root phase (HEWLFROU) with the specified parameters and the default member name.	DMSLKD	CALL	
7 The command FILEDEF * CLEAR is issued to cancel all the file control blocks.	DMSLKD	CALL	
If the return code from the linkage editor is not zero, one of the following messages is displayed. DMSLKD004W WARNING ERROR MESSAGES ISSUED DMSLKD008W ERROR MESSAGES ISSUED DMSLKD012W SEVERE ERROR MESSAGES ISSUED DMSLKD016W TERMINAL ERROR MESSAGES ISSUED		PROCERR	
Control then returns to CMS, with the return code in register 15.		EXIT	

Diagram 9-8. DMSLKD--LKED Command Processor



1 Process the command line.

2 If filename not specified, find and read dump spool file.

3 Define files.

A. Set up the FILEDEF commands.

B. Check that the dump file exists.

C. Execute FILEDEF commands.

4 Call the OS dump program.

5 Return to CMS.

Notes	Module	Label	Ref
1 If the second parameter in the input line starts with DUMP, the name of the CMS file is saved in the output FSCB. The appropriate options are marked in the OPTLIST. If there are no options specified, FORMAT, no MNEMONIC, and no ERASE are assumed. If an invalid option is specified, the following message is generated DMKRND863E INVALID PARAMETER 'xxxxxxxx' and control returns to CMS with a return code of 24.	DMKRND	NCPDUMP TESTOPT	
2 If the name of a CMS dump file was not specified, DMKRND assumes the dump file is in the reader. The filename of the output file is set to DUMP00 through DUMP09 and the STATE macro is issued until a dump file is found. If an available name is not found, the following message is generated. DMKRND851I TEN DUMP FILES ALREADY EXIST and control returns to CMS with a return code of 22.	DMKRND	LOOKLOOP	
The reader is spooled class E and the spool file is read via a DIAGNOSE instruction. The records are deblocked and written to the CMS dump file. The read/write loop continues until the real spool file DIAGNOSE instruction returns a nonzero return code. When the end of file is reached, the message 'DUMPnn NCPDUMP' FILE CREATED is issued, the spool file is closed, and processing continues. If the reader was empty or if a read error occurs, an error message is issued. DMKRND853I NO DUMP FILES EXIST DMKRND850I UNABLE TO READ DUMP FROM READER		READNXT DUMPWRT	
3			
A. The name of the CMS dump file is put in the SYSUT2 and SYSIN FILEDEFs and in the control statement skeleton for the IPLDUMP processor.	DMKRND	STRTDUMP	
B. The STATE macro is issued to check that the CMS dump file exists. If an error is returned, the following message is generated DMKRND861E FILE 'DUMPnn NCPDUMP' NOT FOUND The SYSIN record is created, using the specified user options, any old SYSIN file is erased, and the new SYSIN file is written to the DUMPnn SYSIN file. If the record cannot be written, the message DMKRND870I UNABLE TO CREATE CONTROL FILE FOR IPLDUMP is issued and control returns to CMS.			
C. The following commands are issued to simulate an OS interface. FILEDEF SYSUT2 DISK DUMPnn NCPDUMP A1 (XTENT 513 NOCHANGE FILEDEF SYSIN DISK DUMPnn SYSIN A1 FILEDEF SYSPRINT PRINTER		LINKDMP	
4 DMKRND loads register 1 with the address of a dummy parameter list and links to IFLDUMP. If the return code from IFLDUMP is not zero, it is passed to CMS.	DMKRND		
5 If the return code from IFLDUMP is zero and ERASE has been requested, the DUMPnn file is erased, and the following message is generated 'DUMPnn NCPDUMP' FILE ERASED	DMKRND		

Diagram 9-9. DMKRND--NCPDUMP Command Processor

Program Organization

This section describes the following 3704/3705 command processing modules:

- DMKRND--NCPDUMP command processor
- DMSARN--ASM3705 command processor (for NCP/VS Release 2 and 3 Assembler)
- DMSARX--ASM3705 command processor (for NCP/VS Release 4 Assembler)
- DMSGRN--GEN3705 command processor
- DMSLKD--LKED command processor
- DMSNCP--SAVENCP command processor

DMKRND

The interface to the OS/360 3705 dump program.

Entry Point
DMKRND

Attributes
Runs in a CMS virtual machine

Entry Conditions

R1:	Address of parameter list
R13:	Address of savearea
R14:	Return address
R15:	CSECT base register

Register Usage

R0-10:	Work registers
R11:	Address of FSCBDSECT
R12:	CSECT base register
R13:	Address of savearea
R14:	Linkage register
R15:	Return code

Call to Other Routines
IFLDUMP To format and print the dump

External References
None

Data Areas
FSCB

Exit Conditions

R12:	CSECT base address
R13:	Address of input savearea
R14:	Return address
R15:	Return code

DMSARN

The interface between CMS and the 3704/3705 Assembler (IFKASM).

Entry Points

DMSARN	To process the ASM3705 command.
ASMHAND	To handle any I/O activity pertaining to the SYSUT2 file during the assembly.

Attributes
Disk resident

Entry Conditions

At DMSARN

R1:	Address of the parameter list
R14:	Return address
R15:	Address of the entry point

At ASMHAND

R1:	Address of the DECB
R2:	Address of the DCB
R8:	Address of the CPSECT
R11:	Address of the FCBSECT
R14:	Return address
R15:	Address of the entry point

Register Usage

R0-1:	Work registers
R3:	Base register
R4-5:	Work registers
R6:	Return address to caller
R7-9:	Work registers
R10:	Constant 8
R12-13:	Work registers
R14:	Linkage register
R15:	Error code

Calls to Other Routines

DMSERSA	To erase old files
DMSSMNE	To initialize storage pointers
DMSSTTA	To locate the file
IFKASM	To assemble the 3704/3705 control program

External References

FREEMAIN	To return free storage
GETMAIN	To obtain free storage
NUCON	The nucleus constant area
TYPE	To send messages to the terminal

Data Areas
None

Exit Conditions
Contents of register 15 indicate results of processing.

Return Code	Meaning
0	No errors
4	Minor errors detected during assembly, successful program execution is probable
8	Errors detected during assembly, unsuccessful program execution is possible
12	Serious errors detected during assembly, unsuccessful execution is probable
16	Critical errors detected during assembly, unsuccessful execution is probable
20	Catastrophic errors detected during assembly, partial or complete assembly canceled.
24	Invalid option, no filename
28	File not found
32	Invalid record length for ASM3705 file
36	No read/write disks accessed

DMSARX

The interface between the ASM3705 command and the 3704/3705 Assembler (CWAX00).

Entry	Points
DMSARX	
ASMPROC	SYSUT1 processing routine
TERMPROC	Terminal output processing routine.

Attributes

Executes in user area

Entry Conditions

R1: Address of the parameter list
R14: Return address
R15: Address of the entry point (DMSARX)

Register Usage

R0	NUCON addressability
R1	Address of all PLISTs
R2	Work register
R3	Work register
R4	GETMAIN/FREEMAIN amount
R5	Work register
R6	GETMAIN/FREEMAIN address
R7	ASMPROC address
R8	Work register
R9	Work register
R10	Linkage register
R11	FCB address during ASMPROC
R12	Base register
R13	Save area address
R14	Return register from calls
R15	Assembler root address and return error code

Calls to Other Routines

DMSCRD	Read SYSPARM from console
DMSCWR	Display SYSPARM message to console
DMSFLD	FILEDEF all assembler files
DMSFNS	Close all assembler files
DMSKEY	Control nucleus protect key
DMSERR	Display all error messages
DMSERS	Erase old assembler files
DMSSLN	Load the assembler phases
DMSSMN	Control storage pointers (GETMAIN/FREEMAIN)
DMSSTT	Verify disk file existence
DMSLADAD	SET/RESET the FST chain for auxiliary directory
CWAX00	3705 assembler (XF) root segment

External References

ADT
CMSCB
DMSARD
FSTB
IO
NUCON

Data Areas

DDNAME	Names of CMS ddnames for assembler
OPTLIST	Option list passed to the assembler
OPDEF	(Macro label) names and abbreviations of all options
PARAMLST	Parameter list for assembler
UTENTRY	In-core SYSUT1 record area
UTHEAD	Header area for in-core records
OPTAB$	List of pointers to option table entries
SAVEAREA	SAVEAREA

Exit Conditions

NORMAL

GPR15=0	No error

ERROR

GPR15=24	Invalid option, no filename specified
GPR15=28	File not found
GPR15=32	File not fixed, 80 char. records
GPR15=36	No read/write disks accessed
GPR15=40	Fileid conflict, device invalid for input

Return Code	Meaning
0	No errors
4	Minor errors detected during assembly, successful program execution is probable
8	Errors detected during assembly, unsuccessful program execution is possible
12	Serious errors detected during assembly, unsuccessful execution is probable

16	Critical errors detected during assembly, unsuccessful execution is probable
20	Catastrophic errors detected during assembly, partial or complete assembly canceled
24	Invalid option, no filename
28	File not found
32	Invalid record length for ASM3705 file
36	No read/write disks accessed

DMSGRN

Edits the Stage 2 input for the 3704/3705 control program generation, builds the 3704/3705 assembler files and linkage editor text files, and builds an EXEC macro file.

Entry Point

DMSGRN

Attributes

Runs in a CMS virtual machine

Entry Conditions

R1: Address of the input parameter list
R13: Address of the savearea
R14: Return address
R15: CSECT base address

Register Usage

R0-10: Work registers
R11: Base register 2
R12: Base register 1
R13: Address of the savearea
R14: Linkage register
R15: Return code

Calls to Other Routines

None

External References

None

Data Areas

FSCB

Exit Conditions

R12: Base address
R13: Address of input savearea
R14: Return address
R15: Return code

DMSLKD

The interface to the OS/VS1 Linkage Editor.

Entry Point

DMKSLKD

Attributes

Reusable, disk resident

Entry Conditions

R1: Address of input parameter list

Register Usage

R0-11: Work registers
R12: Base register
R13: Address of savearea
R14-15: Work registers

Calls to Other Routines

DMSSTT To get a copy of an FST
DMSERS To delete a file from disk
DMSLADW To find a read/write disk
DMSFLD To establish file definitions for OS simulation
HEWLFROU To link edit text files

External References

NUCON The nucleus constant area
ADTSECT The active disk table
FSTSECT The file status table

Data Areas

ADT (Active Disk Table)

Exit Conditions

Contents of register 15 indicate results of processing

Return Code	Meaning
0-16	Linkage editor return codes
20	Invalid file ID character
24	No filename specified, missing operand on LIBE or NAME option, or invalid parameter
28	File not found
32	File not fixed 80-byte records
36	No read/write disk accessed or disk not accessed

DMSNCP

Reads a 3705 ccntrol program module (EP or NCP) in OS load module format and writes a page-format core-image copy on the VM/370 system volume.

Entry Point
SAVENCP

Attributes
Serially reusable, executes in a CMS virtual machine

Entry Conditicns
R1: Address of the input parameter list

Register Usage
R0: Work register
R1: Address of parameter list and wcrd register
R2: Pcinter to input record and work register
R3: Length of input record and work register
R4-6: Work registers
R10: Address of the input file DCB during the read, then the address of the ccntrol program core image.
R11: Address of the CCPARM parameter list
R12: Base register
R13: Address of the savearea
R14: Linkage register
R14: Linkage and work register

Calls to Other Routines
DMKSNC via Diagnose Code X'50' to write the core image of the 3704/3705 ccntrol program and parameters on disk

External References
None

Data Areas
CCPARM

Exit Conditions
R15: Return code

Directory

This section contains two types of directories:

- Module Directory (Figure 9-2) is a list of the CP and CMS modules that process the commands that generate the 3704/3705 control program and process the 3704/3705 storage dumps.

- Label Directories (Figures 9-3 through 9-8) list the major labels in each of the command processors. In addition to the label, the module (if more than one is involved), associated method of operation diagram, and a brief description are included in the list.

Module	Description
DMKRND	NCPDUMP command processor.
DMSARN	ASM3705 command processor.
DMSARX	ASM3705 command processor.
DMSGRN	GEN3705 command processor.
DMSLKD	LKED command processor.
DMSNCP	SAVENCP command processor.

Figure 9-2. Module Directory for 3704/3705 Command Processors

THE NCPDUMP COMMAND PROCESSOR (DMKRND)

Label	Diagram	Description
DUMPWRT	9-9	Writes the output file.
LINKDMP	9-9	Links to the OS dump service program, IFLDUMP.
LOOKLOOP	9-9	Checks the reader for a valid CMS dump file.
NCPDUMP	9-9	Starts processing the NCPDUMP command.
READNXT	9-9	Reads the dump spool file.
STRTDUMP	9-9	Builds the control file for the IFLDUMP processing routine.
TESTOPT	9-9	Processes the options on the NCPDUMP command line.

Figure 9-3. The NCPDUMP Command Processor (DMKRND) Label Directory

THE ASM3705 COMMAND PROCESSOR (DMSARN)

Label	Diagram	Description
CONTINUE	9-6	Erases old files and gets enough storage for the assembler to execute in.
DMSARN	9-6	Entry point for the ASM3705 command processor.
LIST2	9-6	Calls the 3705 Assembler (IFKASM).
NOERASE	9-6	Issues FILEDEFs for the necessary assembler files.
RETURN	9-6	Returns control to CMS.
SQUEEZE	9-6	Checks that the input file exists.
SUIT15	9-6	If running in a batch machine, sends ASSEMBLING filename A1 message.
SUIT17	9-6	Finds a read/write disk for writing text and listing files.
SUIT19	9-6	Closes the output files and erases the utility files.
SUIT25	9-6	Checks the format of the input file.

Figure 9-4. The ASM3705 Command Processor (DMSARN) Label Directory

THE ASM3705 COMMAND PROCESSOR (DMSARX)

Label	Diagram	Description
ERASE	9-7	Erases old files.
DMSARX	9-7	Entry point for the ASM3705 command processor.
FILEDEF	9-7	Issues FILEDEFs for the necessary assembler files.
FINDRW	9-7	Finds a read/write disk for writing text and listing files.
LOADASM	9-7	Load the 370X Assembler root.
OPTSCN	9-7	Validates command line.
RETURN	9-7	Returns control to CMS.
VERIFY	9-7	Checks that the input file exists.

Figure 9-5. The ASM3705 Command Processor (DMSARX) Label Directory

Label	Diagram	Description
ASMSTAK	9-4	Stacks the required 3705 Assembler commands in the Stage 2 EXEC macro file.
ASMSTAK2	9-4	Puts the name of the output assembler file in the ASM3705 and EDIT commands.
ASMSTAK4	9-4	Puts the SYSPUNCH membername in the TXTLIB command.
ASMSTAK6	9-4	Puts the number of commands and the address of the first command into registers 1 and 2.
CLOSTACK		Builds the SAVENCP command.
EDITIN		Edits the input records for keywords.
FINDASM	9-3	Checks for assembler input.
FINDIEWL	9-3	Checks for linkage editor input.
FSCBCLOS	9-4	Closes the output file.
FSCBRD		Reads the input file.
FSCBWT		Writes the output file.
GENMSG		Generates error messages.
IEWL	9-5	Main processing routine for generating linkage editor commands.
IEWLENT	9-5	Scans for the keyword ENTRY.
IEWLJCLA	9-5	Edits the //EXEC statement.
IEWLJCL2	9-5	Scans for the //SYSLIN statement.
IEWLSEOF	9-5	Branches and links to FSCBCLOS to close the linkage editor output file.
IEWLSIN		Processes SYSLIN information.
IEWLSN10	9-5	Branches and links to FSCBRD to read the linkage editor input file.
IFKASM	9-4	Main processing routine for generating 3705 assembler files.
IFKASMA0	9-4	Branches and links to the FSCBRD and FSCBWT routines to read the input file and write the output file.
IFKASMK0	9-4	Branches and links to the FSCBCLOS routine to close the output assembler files.
IFKASM10	9-4	Scans for the SYSPUNCH statement.

Figure 9-6. The GEN3705 Command Processor (DMSGRN) Label Directory (Part 1 of 2)

Label	Diagram	Description
IFKASM34	9-4	Scans for the DSN= or DSNAME= keyword on the SYSPUNCH statement.
IFKASM40	9-4	Scans for the SYSIN statement.
LKDSTACK	9-4	Builds the LKED commands and the FILEDEF for their file.
LKDSTAK1	9-5	Loads registers 1 and 2 with the number of commands and the address of the first linkage editor command.
OPTEND	9-3	Checks that the input file exists.
OPTIONS1	9-3	Scans the input options.
PRIMEDIT	9-3	Scans for a valid //EXEC statement.
PROCEND1	9-3	Closes the EXEC file.
PROCEND2	9-3	Adds the SAVENCP command to the EXEC macro file.
PROCWT		Writes commands to the Stage 2 EXEC processor file.
RETURN1	9-3	Returns control to CMS.
STACK30	9-3	Writes the linkage editor and assembler statements to the EXEC macro file.
START	9-3	Starts the GEN3705 command processing.
WRTSIN	9-4	Branches and links to the FSCBWT routine to write the linkage editor output file.

Figure 9-6. The GEN3705 Command Processor (DMSGRN) Label Directory (Part 2 of 2)

THE LKED COMMAND PROCESSOR (DMSLKD)

Label	Diagram	Description
CALL	9-8	Calls the OS/VS1 Linkage Editor (HEWLFROU).
DMSLKD	9-8	Entry point for the LKED command processor.
EXIT	9-8	Returns control to CMS.
OUTLOOP2	9-8	Processes the command options.
PROCERR	9-8	Processes the error messages.
PRTDEF	9-8	Sets up the file definition for the printer.

Figure 9-7. The LKED Command Processor (DMSLKD) Label Directory

Label	Diagram	Description
CESDCHVT	9-1	Finds the channel vector table.
CESDENT	9-1	Saves the entry point.
CHEKVMV	9-2	Checks that the specified options are compatible.
CLOSE	9-1	Closes the input file.
CONTROL	9-1	Moves the text records from the input buffer to the core image buffer.
ENDPARMS	9-1	Opens the input file and searches for the member.
ERR21	9-1	Checks for the entry point record.
ERR66	9-1	Checks for premature end of file or invalid control records.
EXIT	9-1	Returns control to CMS.
SAVECCP	9-1	Issues the Diagnose X'5C' instruction to have DMKSNC do the actual saving.
SAVENCP	9-1	Entry point for the SAVENCP command processor.
SCANCEP	9-2	Updates the CCPARM parameter list for EP and PEP control programs.
SCANDEV		Scans for devices.
SCANLINE		Scans for teleprocessing lines.
SCANNCP	9-2	Updates the CCPARM parameter list for NCP and PEP control programs.

Figure 9-8. The SAVENCP Command Processor (DMSNCP) Label Directory

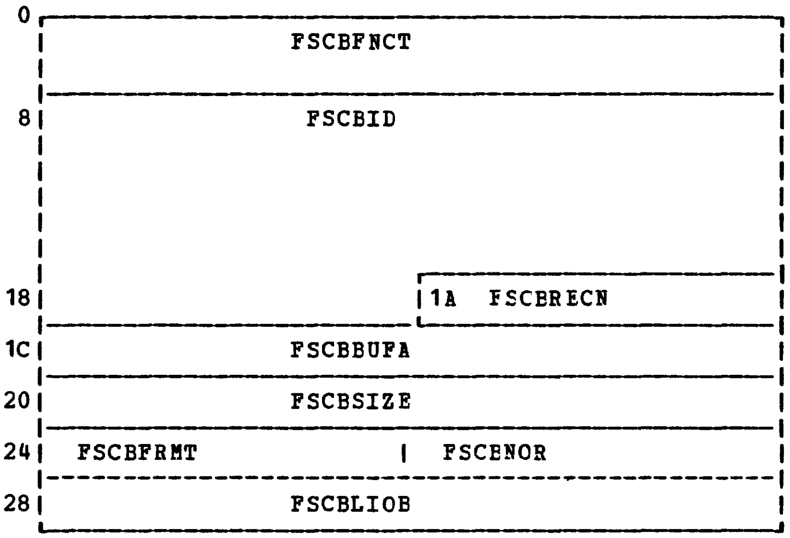
Data Areas

The following data areas are used by the 3704/3705 command processor modules:

- Active Disk Table (ADT)
- Communications Controllers Parameter List (CCPARM)
- File System Control Block (FSCB)
- Input/Output Block (IOBLOK)
- Network Interface Control Block (NICBLOK)
- Real Device Block (RDEVBLOK)
- Spool File Block (SFBLOK)
- Virtual Machine Block (VMBLOK)

All the above data areas except the FSCB are described in the VM/370 Data Areas and Control Blocks Logic. The FSCB is described in Figure 9-8.

FILE SYSTEM CONTROL BLOCK



Displacement					
Hex	Dec	Field Name			Description
0	0	FSCBFNCT	DS	CL8	Control field for I/O function
8	8	FSCBID	DS	OCL18	File Identifier
8	8	FSCBFN	DS	CL8	Filename
10	16	FSCBFT	DS	CL8	Filetype
18	24	FSCBFM	DS	CL2	Filemode
1A	26	FSCBRECN	DS	H	Relative record number
1C	28	FSCBBUFA	DS	A	Buffer address
20	32	FSCBSIZE	DS	F	Buffer size
24	36	FSCBFRMT	DS	CL2	File format
26	38	FSCBNOR	DS	OH	Number of records to be read
28	40	FSCBLIOB	DS	A	

Figure 9-9. File System Control Block (FSCB)

Diagnostic Aids

The following figures list the messages and abnormal termination codes issued by the CMS 3704/3705 command processors.

Figure 9-10 lists the messages issued by the NCPDUMP command processor (DMKRND).

Figures 9-11 and 9-12 list the messages issued by the ASM3705 command processor (DMSARN and DMSARX).

Figure 9-13 lists the messages issued by the GEN3705 command processor (DMSGRN).

Figure 9-14 lists the messages issued by the LKED command processor (DMSLKD).

Figure 9-15 lists the messages issued by the SAVENCP command processor (DMSNCP).

THE NCPDUMP COMMAND PROCESSOR (DMKRND)

Message Code	Label	Diagram	Message Text
DMKRND850I	DUMPWRT	9-9	UNABLE TO READ DUMP FROM READER (Return Code = 21)
DMKRND851I	LOOKLOOP	9-9	TEN DUMP FILES ALREADY EXIST (Return Code = 22)
DMKRND852I			FATAL I/O ERROR WRITING DUMP
DMKRND853I	DUMPWRT	9-9	NO DUMP FILES EXIST (Return Code = 23)
DMKRND861E	STRTDUMP	9-9	FILE 'DUMPnn NCPDUMP' NOT FOUND (Return Code = 28)
DMKRND863E	TESTOPT	9-9	INVALID PARAMETER - 'xxxxxxxx' (Return Code = 24)
DMKRND870I	STRTDUMP	9-9	UNABLE TO CREATE CONTROL FILE FOR IFLDUMP (Return Code = 16)
	DUMPWRT	9-9	'DUMPnn NCPDUMP' FILE CREATED
	LINKDMP	9-9	'DUMPnn NCPDUMP' FILE ERASED

Figure 9-10. The NCPDUMP Command Processor (DMKRND) Error Messages

THE ASM3705 COMMAND PROCESSOR (DMSARN)

Message Code	Label	Diagram	Message Text
DMSARN001E	DMSARN	9-6	NO FILENAME SPECIFIED
DMSARN002E			FILE 'fn ASM37C5' NOT FOUND
DMSARN003E			INVALID OPTION 'xxxxxxxx'
DMSARN004W	RETURN	9-6	WARNING MESSAGES ISSUED
DMSARN006E			NO READ/WRITE CISK ACCESSED
DMSARN007E	SUIT25	9-6	FILE 'fn ft' IS NOT FIXED, 80-CHAR. RECCRDS
DMSARN008W	RETURN	9-6	ERROR MESSAGES ISSUED
DMSARN012W	RETURN	9-6	SEVERE ERROR MESSAGES ISSUED
DMSARN016W	RETURN	9-6	TERMINAL ERROR MESSAGES ISSUED
DMSARN109S			VIRTUAL STORAGE CAPACITY EXCEEDED

Figure 9-11. The ASM3705 Command Processor (DMSARN) Error Messages

THE ASM3705 COMMAND PROCESSOR (DMSARX)

Message Code	Label	Diagram	Message Text
DMSARX001E	OPTSCN	9-7	NO FILENAME SPECIFIED
DMSARX002E	NEWFILE	9-7	FILE 'fn ASM37C5' NOT FOUND
DMSARX003E	OPTSCN	9-7	INVALID OPTION 'option'
DMSARX007E	FINDRW	9-6	NO READ/WRITE CISK ACCESSED
DMSARX007E	STATASM	9-7	FILE 'fn ASM37C5' IS NOT FIXED, 80-CHAR. RECORDS
DMSARX038E	DOFDEF	9-7	FILEID CONFLICT FOR CCNAME 'ASM3705'
DMSARX052E	MOVEKEY	9-7	MORE THAN 1C0 CHARS. OF OPTICNS SPECIFIED
DMSARX070E	DMSARX	9-7	INVALID PARAMETER 'parameter'
DMSARX074E	DMSARX	9-7	ERROR [RE]SETTING AUXILIARY DIRECTCRY
DMSARX075E	NOTDSK	9-7	DEVICE 'device' INVALID FOR INPUT

Figure 9-12. The ASM3705 Command Processor (DMSARX) Error Messages

THE GEN3705 COMMAND PROCESSOR (DMSGRN)

Message Code	Label	Diagram	Message Text
DMSGRN002E	OPTEND	9-3	FILE 'fn ft' NOT FOUND
DMSGRN003E	OPTIONS1	9-3	INVALID OPTION 'option'
DMSGRN007E			FILE 'fn ft' IS NOT FIXED, 80 CHAR. RECORDS
DMSGRN048E	OPTEND	9-3	INVALID MODE 'fm'
DMSGRN054E			INCOMPLETE FILE ID SPECIFIED
DMSGRN078E	PRIMEDIT IFKMAS40 IEWLJCL2	9-3 9-4 9-5	INVALID CARD IN INPUT FILE 'xxx... x'

Figure 9-13. The GEN3705 Command Processor (DMSGRN) Error Messages

THE LKED COMMAND PROCESSOR (DMSLKD)

Message Code	Label	Diagram	Message Text
DMSLKD001E	DMSLKD	9-8	NO FILE NAME SPECIFIED
DMSLKD002E	OUTLOOP2	9-8	FILE 'fn ft' NOT FOUND
DMSLKD004W	PROCERR	9-8	WARNING ERROR MESSAGES ISSUED
DMSLKD005E	OUTLOOP2	9-8	NO 'option' SPECIFIED
DMSLKD006E	PRTDEF	9-8	NO READ/WRITE DISK ACCESSED
DMSLKD008E	OUTLOOP2	9-8	FILE 'fn ft' IS NOT FIXED, 80 CHAR. RECORDS
DMSLKD008W	PROCERR	9-8	ERROR MESSAGES ISSUED
DMSLKD012W	PROCERR	9-8	SEVERE ERROR MESSAGES ISSUED
DMSLKD016W	PROCERR	9-8	TERMINAL ERROR MESSAGES ISSUED
DMSLKD080E			INVALID PARAMETER 'parameter'

Figure 9-14. The LKED Command Processor (DMSLKD) Error Messages

Message Code	Label	Diagram	Message Text
DMSNCP001E	SAVENCP	9-1	NO FILENAME SPECIFIED (Return Code = 24)
DMSNCP002E	ENDPARMS	9-1	FILE 'fn ft fm' NOT FOUND (Return Code = 28)
DMSNCP003E	SAVENCP TESTOP		INVALID OPTION - 'option' (Return Code = 24)
DMSNCP013E	DMS0001A	9-1	MEMBER 'name' NOT FOUND IN LIBRARY 'fn ft' (Return Code = 4)
DMSNCP021E	CONTROL	9-1	ENTRY POINT 'symbol' NOT FOUND (Return Code = 40)
DMSNCP025E	SCANCEP SCANNCP	9-2	INVALID DATA IN 370X CONTROL PROGRAM (Return Code = 16)
DMSNCP045E	CLOSE		UNSUPPORTED 370X CONTROL PROGRAM TYPE (Return Code = 16)
DMSNCP056E	NOTLAST CLOSE ERR66	9-1	FILE 'fn ft' CONTAINS INVALID RECORD FORMATS (Return Code = 32)
DMSNCP099W	CHEKVMV	9-2	GENERATION PARAMETERS INCOMPATIBLE WITH VM/370 (Return Code = 99)
DMSNCP109S	CONTROL NOTLAST	9-1	VIRTUAL STORAGE CAPACITY EXCEEDED (Return Code = 104)

Figure 9-15. The SAVENCP Command Processor (DMSNCP) Error Messages

Chapter 10. The ZAP Service Program

Introduction

The ZAP service program (DMSZAP) executes under the control of CMS via the ZAP command. It performs three functions for LOADLIB, TXTLIB, and MODULE files residing on direct access storage devices. The functions are:

- Dump
- Verify
- Replace

DUMP

The dump function reads all or part of a specified CSECT, or an entire member or module, formats the dump, and prints it at the system printer (133-character lines, each containing 32 bytes in hexadecimal, plus the translation) or displays it at the terminal (80-character lines, each containing 16 bytes in hexadecimal, plus the translation). If more than one CSECT is dumped, the CSECT name appears before each dump.

VERIFY

The verify function compares specified data with the data at a specified address in a CSECT. If the data is the same, a replace operation (if one is specified) is permitted; otherwise, an error message is issued.

REPLACE

The replace function replaces data at a specified address in a CSECT with the data specified in a control record. The changed record is then written back to the file.

Method of Operation

The method of operation diagrams describe the execution of the ZAP program and show the processing associated with:

- Verifying and replacing data in a CSECT.
- Dumping a CSECT, member, or module.

The relationship of the method of operation diagrams is shown in Figure 10-1.

Diagram 10-1 describes the execution of the ZAP program.

Diagram 10-2 shows the ZAP command and control record processing.

Diagram 10-3 describes the processing of the DUMP function.

Diagrams 10-4 and 10-5 describe the processing for modifying data in a CSECT.

Diagrams 10-6 and 10-7 describe how the proper CSECT is located for dumping or modifying.

Diagram 10-8 shows how a file is read for dumping or modifying.

Diagram 10-9 describes how a dump is printed.

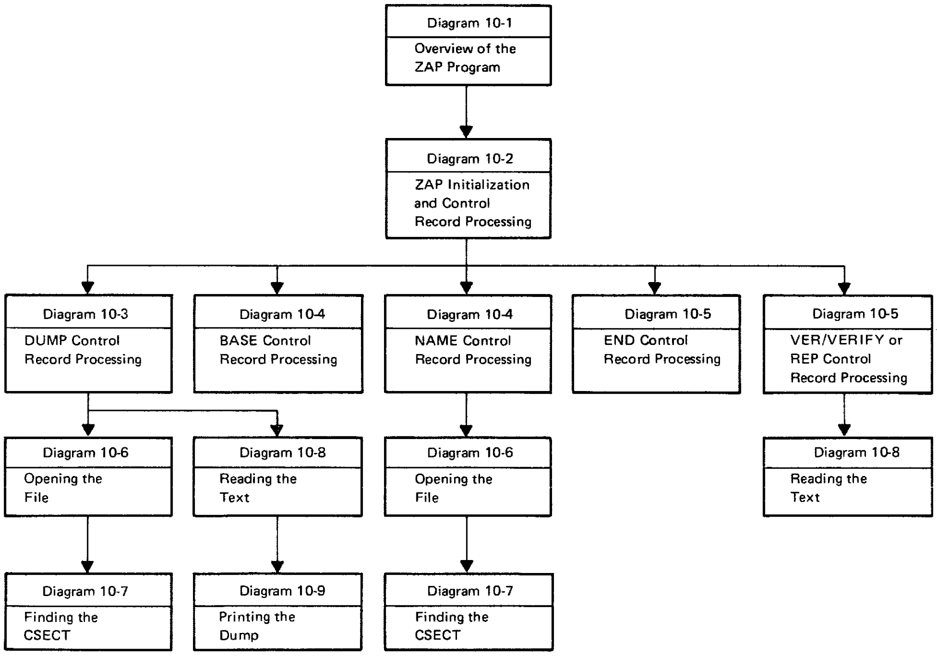
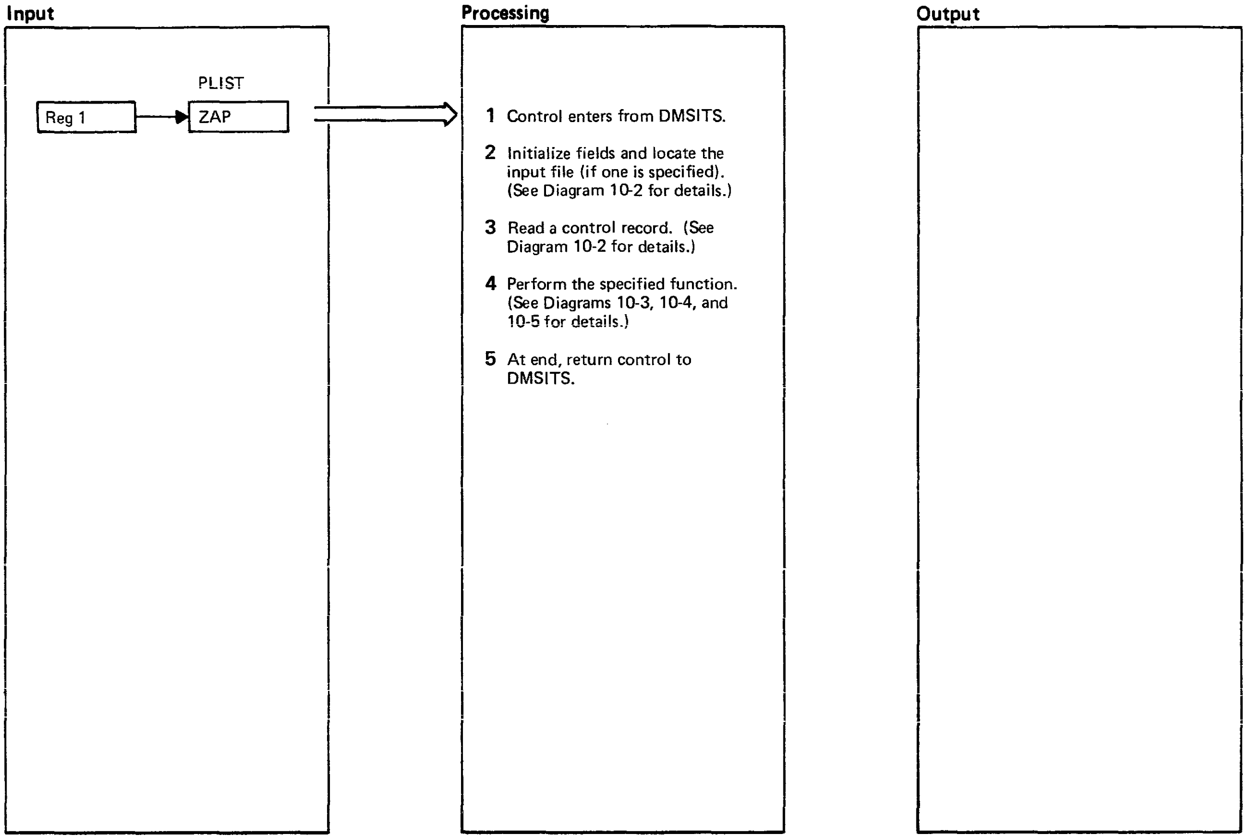


Figure 10-1. Key to the ZAP Program Method of Operation Diagrams



1 Control enters from DMSITS.

2 Initialize fields and locate the input file (if one is specified). (See Diagram 10-2 for details.)

3 Read a control record. (See Diagram 10-2 for details.)

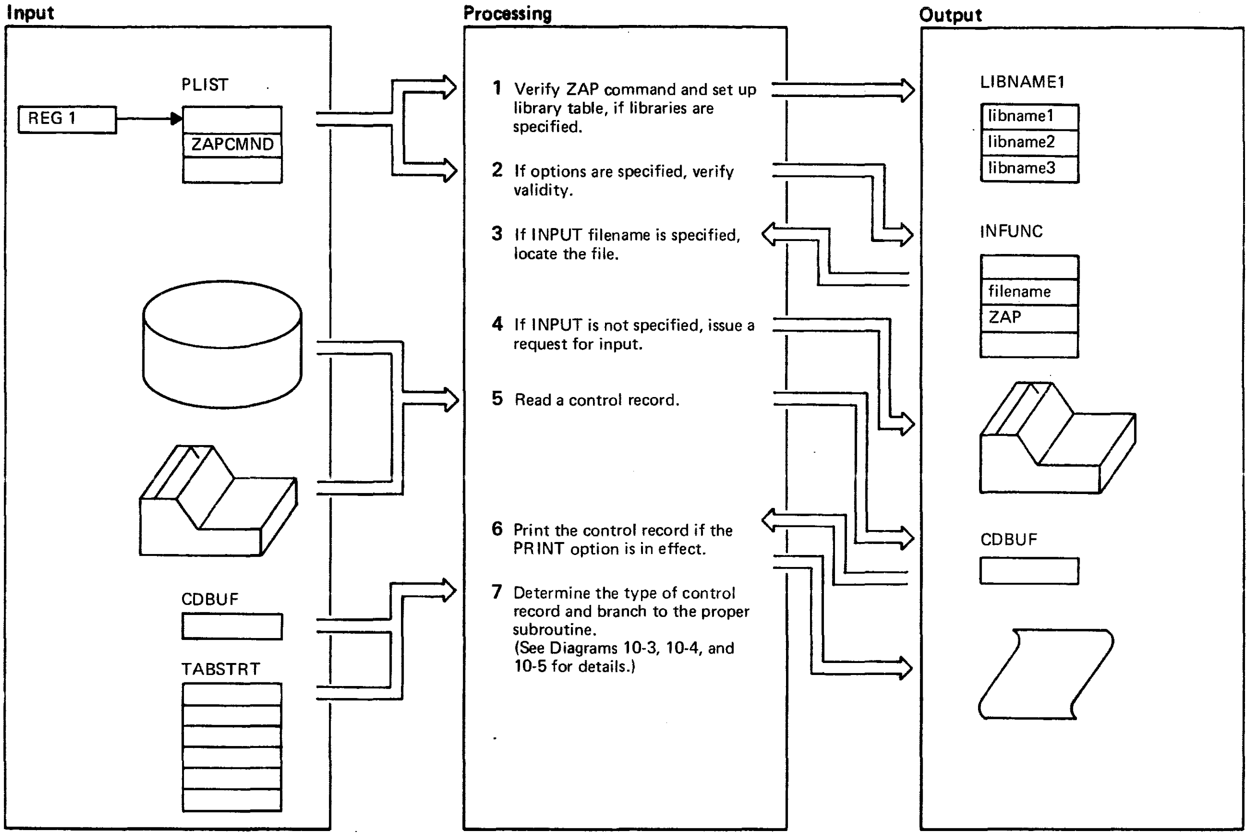
4 Perform the specified function. (See Diagrams 10-3, 10-4, and 10-5 for details.)

5 At end, return control to DMSITS.

Notes	Module	Label	Ref
1 Control enters DMSZAP from DMSITS. Register 1 points to a PLIST that contains the type of file to be operated on, libraries to be used if applicable, and controls for input and output operations.	DMSZAP	DMSZAP	
2 Initialize fields and pointers and verify input and output options. Locate the input file if an input file is specified. Otherwise, request input from the terminal.	DMSZAP	SCANLINE INITOPEN FDEFINP	
3 Read a control record. Find the routine needed to perform the function specified by searching a table of control record keywords.	DMSZAP	READINP	
4 Perform the specified function. At its end, return control to READINP to read another control record.	DMSZAP		
5 When the END control record is read, return control to DMSITS.	DMSZAP		

Notes	Module	Label	Ref

Diagram 10-1. Overview of the ZAP Program



1 Verify ZAP command and set up library table, if libraries are specified.

2 If options are specified, verify validity.

3 If INPUT filename is specified, locate the file.

4 If INPUT is not specified, issue a request for input.

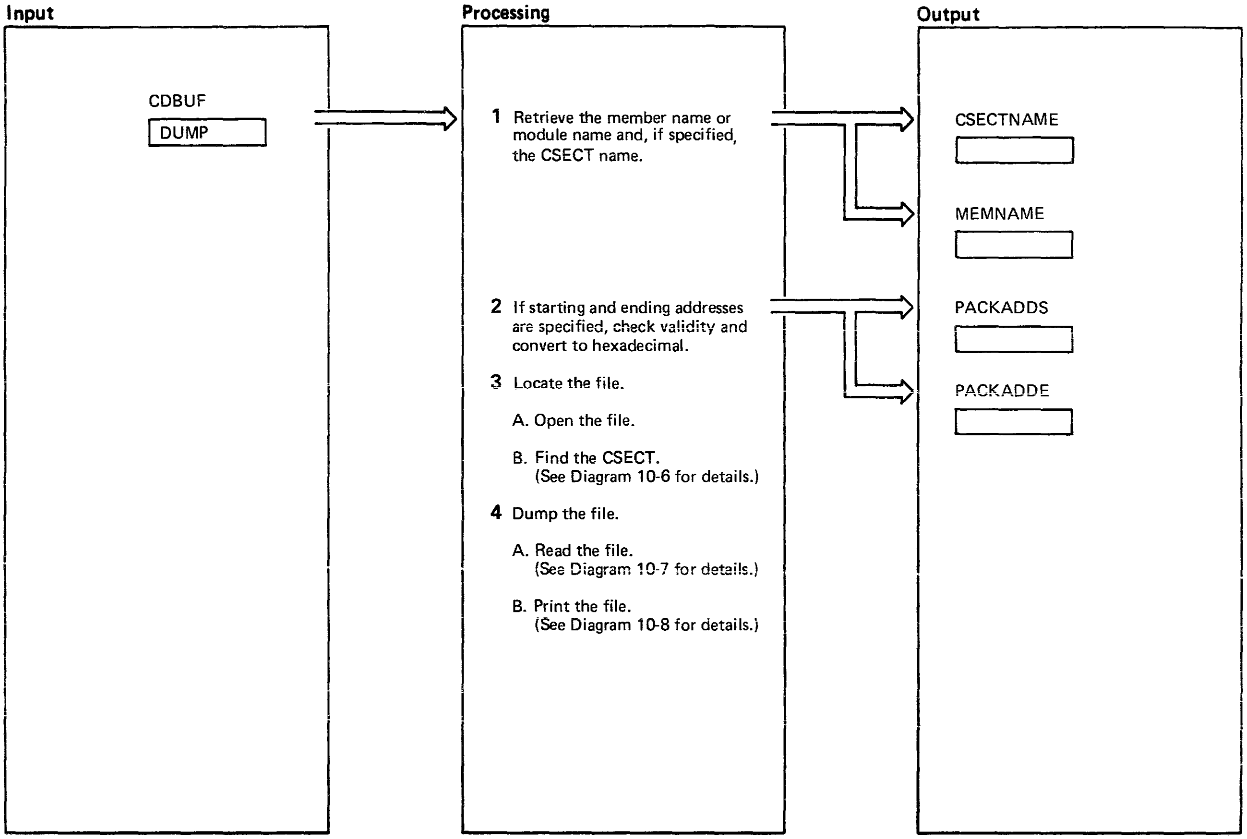
5 Read a control record.

6 Print the control record if the PRINT option is in effect.

7 Determine the type of control record and branch to the proper subroutine. (See Diagrams 10-3, 10-4, and 10-5 for details.)

Notes	Module	Label	Ref
1 Verify the operands in the ZAP command. If TXTLIB or LOADLIB is specified, move the library names (up to three) into LIBNAME1. If no library name was specified, issue the message: DMSZAP001E NO FILENAME SPECIFIED Other messages that may be issued if the command line is in error are: DMSZAP014E INVALID FUNCTION 'function' DMSZAP047E NO FUNCTION SPECIFIED DMSZAP070E INVALID PARAMETER 'param'	DMSZAP	SCANLINE STLIB	
2 If options are specified, check for validity. If mutually exclusive options or invalid options are specified, issue the message: DMSZAP003E INVALID OPTION 'option'	DMSZAP	CHKOPT	
3 If INPUT filename is specified, move filename into INFUNC. Issue STATE to locate the file. If this file cannot be found, the message: DMSZAP002E FILE 'fn ft' NOT FOUND	DMSZAP	INPTOPT FDEFINP	
4 If INPUT is not specified, display ENTER: to request ZAP control records to be entered from the terminal.	DMSZAP	READINP RDCARD	
5 Read the control record either from the terminal (RDCARD routine) or from the specified INPUT file (RDCARD2 routine). Save the control record in CDBUF.	DMSZAP	RDCARD RDCARD2	
6 Print the control record on the SYSOUT printer if the PRINT option is in effect.	DMSZAP	WRCARD	
7 Check the control record for a valid keyword. If the statement is blank or the first character is an asterisk, return control to READINP (step 4). Otherwise, compare the keyword to keyword tables whose formats are: bytes 1-8 keyword bytes 9-12 keyword routine Valid keywords and the diagrams in which their routines are described are: Keyword Diagram DUMP 10-3 NAME 10-4 BASE 10-4 VER 10-5 VERIFY 10-5 REP 10-5 END 10-5 If a match is found, go to the appropriate routine. If no match is found, issue the message: DMSZAP201W INVALID CONTROL RECORD OR NO GO SWITCH SET and return control to READINP (step 4).	DMSZAP	SCANKEY1 TABLOOK NAMFOUND INVEREP	

Diagram 10-2. ZAP Initialization and Control Record Processinq



1 Retrieve the member name or module name and, if specified, the CSECT name.

2 If starting and ending addresses are specified, check validity and convert to hexadecimal.

3 Locate the file.

A. Open the file.

B. Find the CSECT. (See Diagram 10-6 for details.)

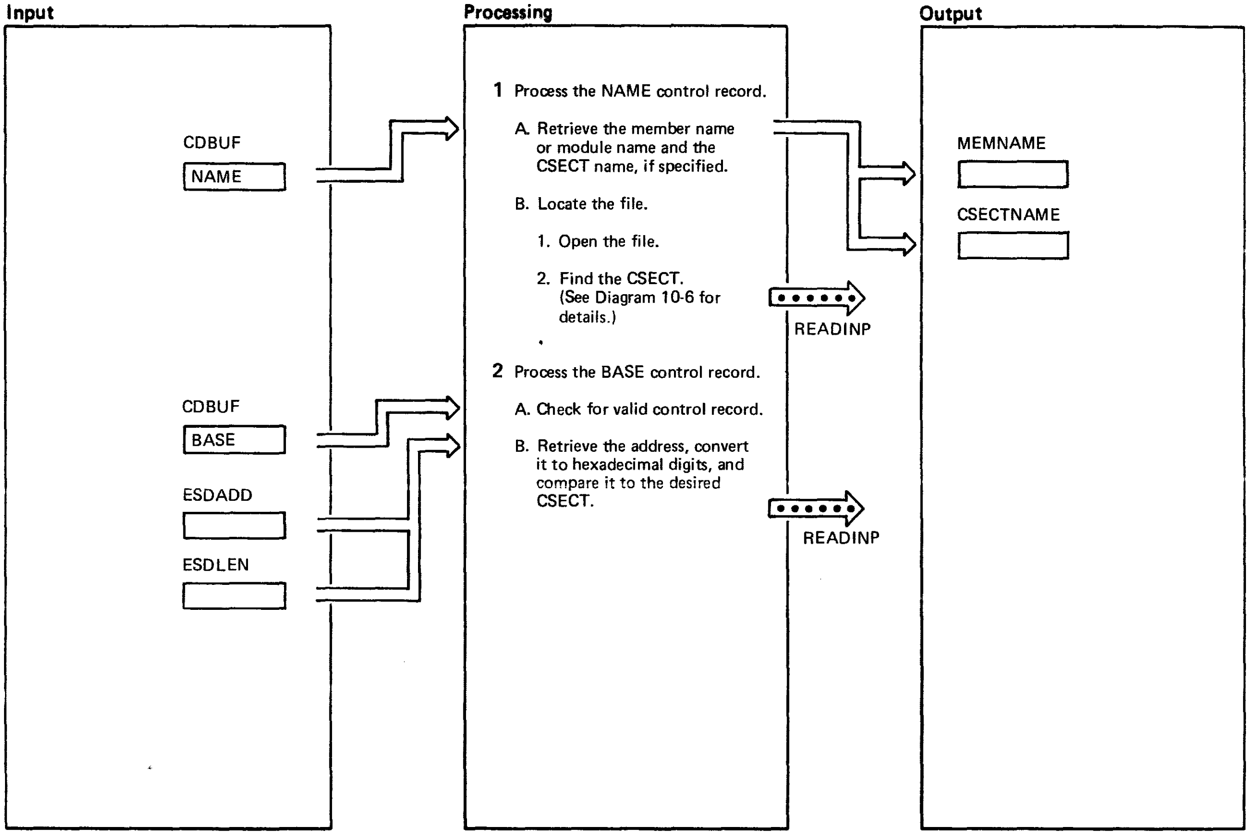
4 Dump the file.

A. Read the file. (See Diagram 10-7 for details.)

B. Print the file. (See Diagram 10-8 for details.)

Notes	Module	Label	Ref
1 Retrieve the member name or module name, if specified, from the control record. If an error is encountered, issue the message DMSZAP201W INVALID CONTROL RECORD OR NO GO SWITCH SET Continue by reading another control record.	DMSZAP	DUMPREC DUMPERR	
2 If starting and ending addresses are specified, retrieve them from the control record, check them for validity, and convert them into hexadecimal digits. If either of the addresses is not an even number of digits, issue the message DMSZAP203W – ERROR – ODD NUMBER OF DIGITS – SET NO GO SWITCH and continue by reading another control record:	DMSZAP	DMPNTALL SCANKEY1 DECODE1 PACKVAL INVEREP2	
3 Go to the open routine (PREOPLIB) to locate the member or module and the CSECT desired.	DMSZAP	DMPCSECT PREOPLIB	
4 Use the starting and ending addresses of the CSECT to determine the length of the dump if not otherwise specified. Go to the read text routine to read the file (RDTXT) and then to the print dump routine (PRTDUMP).	DMSZAP	STSTART GORDTXT RDTXT PRTDUMP	
If all CSECTs are requested, return control to step 3. When the request is satisfied, read another control record (see Diagram 10-2, Step 4).		READINP	

Diagram 10-3. DUMP Control Record Processing



1 Process the NAME control record.

A. Retrieve the member name or module name and the CSECT name, if specified.

B. Locate the file.

1. Open the file.

2. Find the CSECT. (See Diagram 10-6 for details.)

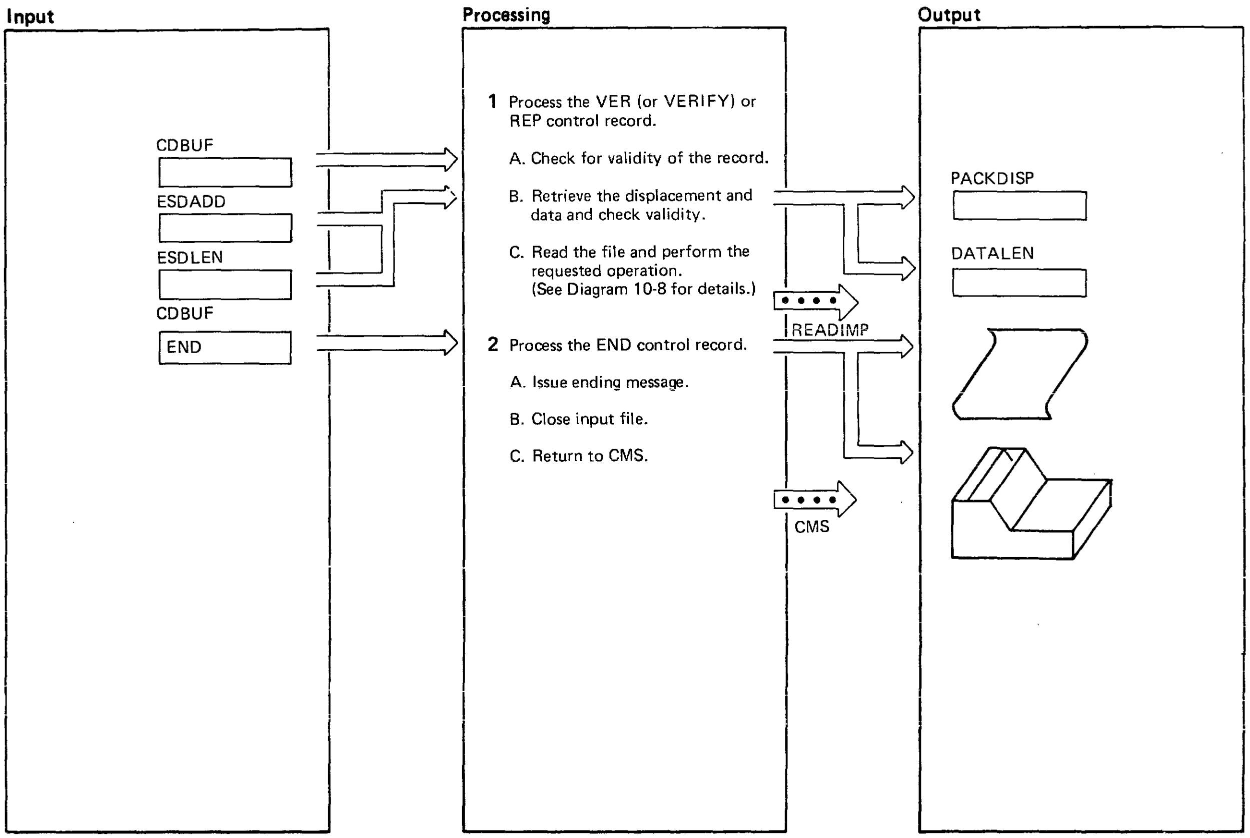
2 Process the BASE control record.

A. Check for valid control record.

B. Retrieve the address, convert it to hexadecimal digits, and compare it to the desired CSECT.

Notes	Module	Label	Ref
1			
A. Retrieve the member name or module name and the CSECT name, if specified, and check for errors. If errors are found, issue the message DMSZAP190W INVALID CONTROL RECORD OR NO GO SWITCH SET Continue by reading another control record.	DMSZAP	NAMEREC INVEREP	
B. If no errors are found, open the specified file and locate the desired CSECT. Continue by reading another control record.	DMSZAP	NOCSECT1 PREOPLIB READINP	
2			
A. Check that the NAME control record has been entered. If not, issue the message DMSZAP190W INVALID CONTROL RECORD OR NO GO SWITCH SET Continue by reading another control record.	DMSZAP	BASEREC INVEREP	
B. Retrieve the BASE address, check it for accuracy, and convert it to hexadecimal.	DMSZAP	CKBASE DECODE1 PACKVAL	
If the address is not an even number of digits, issue the message DMSZAP192W ERROR – ODD NUMBER OF DIGITS – SET NO GO SWITCH and continue by reading another control record.		INVEREP2	
If the file is a MODULE file created with the NOMAP option, accept the BASE address and continue by reading another control record.			
If the file is a LOADLIB or TXTLIB file, or a MODULE file not created with the NOMAP option, compare the BASE address to the CSECT address. If there is a match, continue by reading another control record.		CKBASE1	
If the CSECT address is not equal to the BASE address, issue the message DMSZAP195W BASE VALUE INVALID – SET NO GO SWITCH Continue by reading another control record.		INVEREP2	

Diagram 10-4. NAME and BASE Control Record Processing



1 Process the VER (or VERIFY) or REP control record.

A. Check for validity of the record.

B. Retrieve the displacement and data and check validity.

C. Read the file and perform the requested operation. (See Diagram 10-8 for details.)

2 Process the END control record.

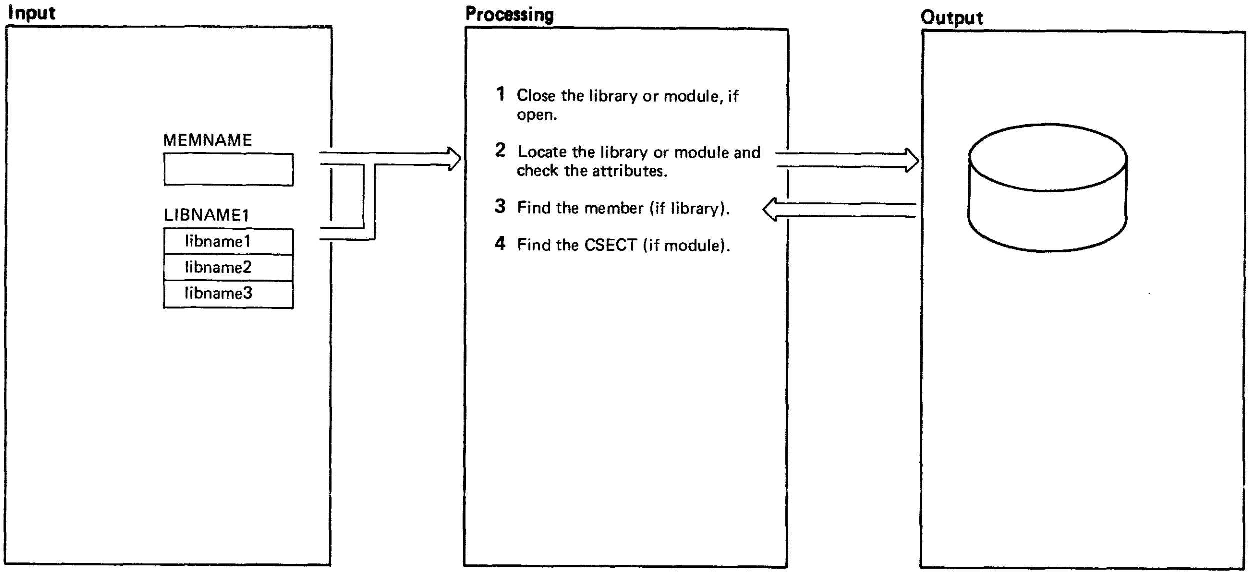
A. Issue ending message.

B. Close input file.

C. Return to CMS.

Notes	Module	Label	Ref
1			
A. If a NAME control record has not been enetered or was invalid, issue the message DMSZAP190W INVALID CONTROL RECORD OR NO GO SWITCH SET and return control to READINP to read another control record. Ignore all VER or REP control records until the next NAME control record is encountered.	DMSZAP	GOODTHRE INVEREP	
If this is a REP control record and the NO GO swtich is on, issue the message DMSZAP193W PRECEDING CONTROL RECORD FLUSHED and return control to READINP to read another control record.		INVEREP2	
B. Check the displacement for validity and convert into hexadecimal digits.	DMSZAP	GOOK SCANKEY1 DECODE1 PACKVAL	
Retrieve the data field, remove commas from the field, and check that the data are an even number of bytes. If not, issue the message DMSZAP192W ERROR – ODD NUMBER OF DIGITS – SET NO GO SWITCH and return control to READINP to read another control record.	DMSZAP	SCANKEY1 CKCOMMA2 CKCOMMA3 INVEREP2	
Convert the data to hexadecimal and add the BASE value to the displacement. Check that the displacement plus the data length will fit within the CSECT. If not, issue the message DMSZAP191W PATCH OVERLAPS – SET NO GO SWITCH and return control to READINP.	DMSZAP	EQLNTH PACKDAT INVEREP2	
C. Go to the RDTXT routine to perform the operation, then return control to READINP.	DMSZAP	GOVER RDTXT	
2			
A. Issue the message DMSZAP750I ZAP PROCESSING COMPLETE	DMSZAP	COMEND INVEREP4	
B. Close the INPUT file, if it is open, and free buffer space.	DMSZAP	CLOSEINP CLRSPCE	
C. Return to CMS.	DMSZAP	NOMORE	

Diagram 10-5. VER/VERIFY or REP and END Control Record Processing



1 Close the library or module, if open.

2 Locate the library or module and check the attributes.

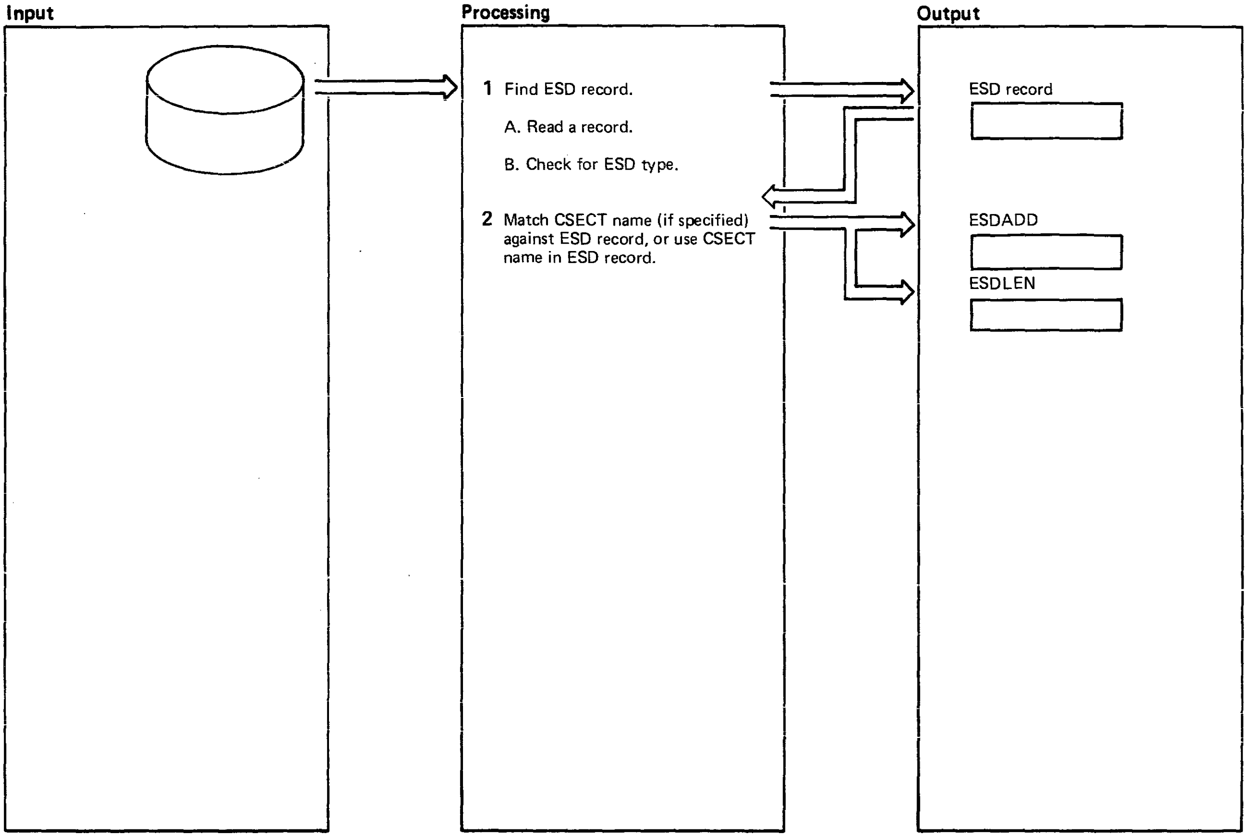
3 Find the member (if library).

4 Find the CSECT (if module).

Notes	Module	Label	Ref
1 Close input module and library files, if open.	DMSZAP	PREOPLIB CLOSELIB	
2 If MODULE was specified, locate the module name and search for the module. If the module is found, check the attributes and if they are valid, go to Step 4. Otherwise, issue one of these error messages:	DMSZAP	STFDEF	
DMSZAP210E FILE 'fn ft' IS ON A READ/ONLY DISK		LIBRO	
DMSZAP208E FILE 'fn ft' IS NOT VARIABLE RECORD FORMAT		LIBNTV	
If the module cannot be found, issue the message		PREOPLB3	
DMSZAP002W FILE 'fn ft' NOT FOUND		PREOPLB5	
and read another control record. Ignore all control records until the next NAME, DUMP, or END control record.		INVEREP2	
If LOADLIB or TXTLIB was specified, locate the first library name and search for the member. If the member is found, check the attributes and if they are invalid, issue one of these messages:		STFDEF	
DMSZAP210E FILE 'fn ft' IS ON A READ/ONLY DISK		LIBRO	
DMSZAP208E FILE 'fn ft' IS NOT VARIABLE RECORD FORMAT		LIBNTV	
DMSZAP007E FILE 'fn ft' IS NOT FIXED, 80 CHAR. RECORDS		FILENTF	
Otherwise, go to Step 3 after issuing the message		MEMFND	
DMSZAP751I MEMBER FOUND IN LIBRARY 'fn'			
If the library cannot be found, issue the message		PREOPLB3	
DMSZAP002W FILE 'fn ft' NOT FOUND		INVEREP2	
and locate the next library name and execute Step 2 again. If none of the libraries specified can be found, issue the message		LIBNTFD1	

Notes	Module	Label	Ref
DMSZAP002E FILE 'fn ft' NOT FOUND			
and terminate processing.		NOMORE	
3 When a library is found, read the first record. If the header record or the pointer to the directory is invalid, issue the message	DMSZAP	OPENFILE PREOPLB4	
DMSZAP056E FILE 'fn ft' CONTAINS INVALID RECORD FORMATS		INVFORM	
Otherwise, locate the directory record and search for the member name. If the file is a CMS-only (not OS) TXTLIB file and the member name cannot be found, search for the CSECT name. If a member name or CSECT name is found, go to the READCESD routine to find a CSECT record.		READLIB CHKMEM CHKCSECT	
4 If the file is a MODULE, compute the length of the module and its starting and ending addresses. Determine if a map is present and, if not, that no CSECT name was specified, then exit. If a CSECT name was specified, issue the message	DMSZAP	CHKLDTBL NOTABLE INVEREP2 CHKLDCST	
DMSZAP246W NO LOADER TABLE PRESENT FOR MODULE 'fn' SET NO GO SWITCH			
then exit. If a module map is present, locate the map record and read it. If the map record cannot be found, issue the message			
DMSZAP056E FILE 'fn ft' CONTAINS INVALID RECORD FORMATS		INVFORM	
Otherwise, locate the CSECT specified or the first CSECT in the map, and determine its length, and return control to caller. If the CSECT specified cannot be found, issue the message		LDRLOOP FNDCLNTH INVEREP2	
DMSZAP194W CSECT NOT FOUND IN 'fn ft' – SET NO GO SWITCH			
and read another control record.			

Diagram 10-6. Opening the File



1 Find ESD record.

A. Read a record.

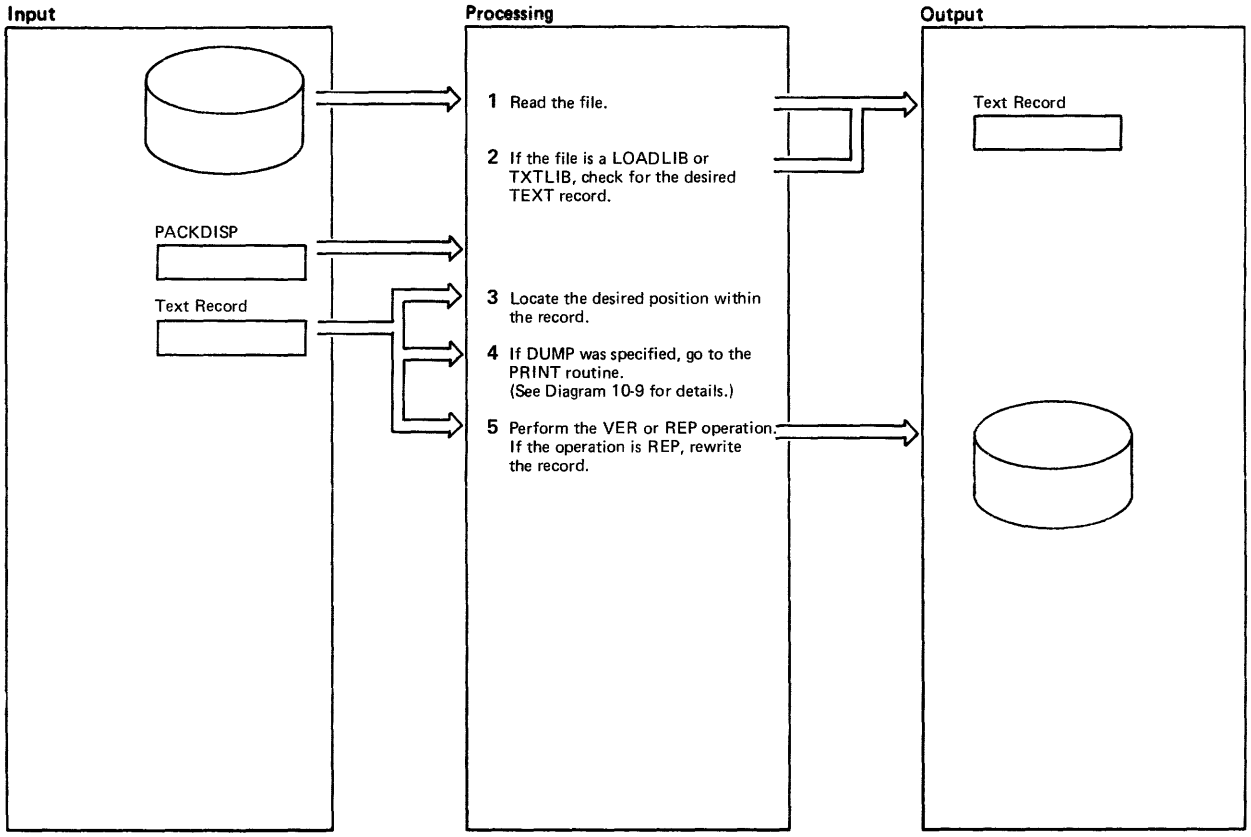
B. Check for ESD type.

2 Match CSECT name (if specified) against ESD record, or use CSECT name in ESD record.

Notes	Module	Label	Ref
1 Read a LOADLIB or TXTLIB member record. Check to see if it is an ESD-type record. If not, re-execute Step 1.	DMSZAP	READCESD TXTESD RDLIB	
2 If a CSECT name was specified in the NAME or DUMP control record, compare it with the CSECT name(s) in the ESD record(s).		SEARCHSD	
If there is a match, save the starting address and length.		CSECTFND	
If there is no match, issue the message DMSZAP194W CSECT NOT FOUND IN 'fn ft' – SET NO GO SWITCH		NOCESD2	
If no CSECT name was specified in the NAME or DUMP control record, use the first CSECT named in an ESD record.			
If ALL was specified in a DUMP control record, use the next CSECT name encountered in an ESD record.			
Control then returns to caller.		MEMEND	

Notes	Module	Label	Ref

Diagram 10-7. Finding the CSECT



1 Read the file.

2 If the file is a LOADLIB or TXTLIB, check for the desired TEXT record.

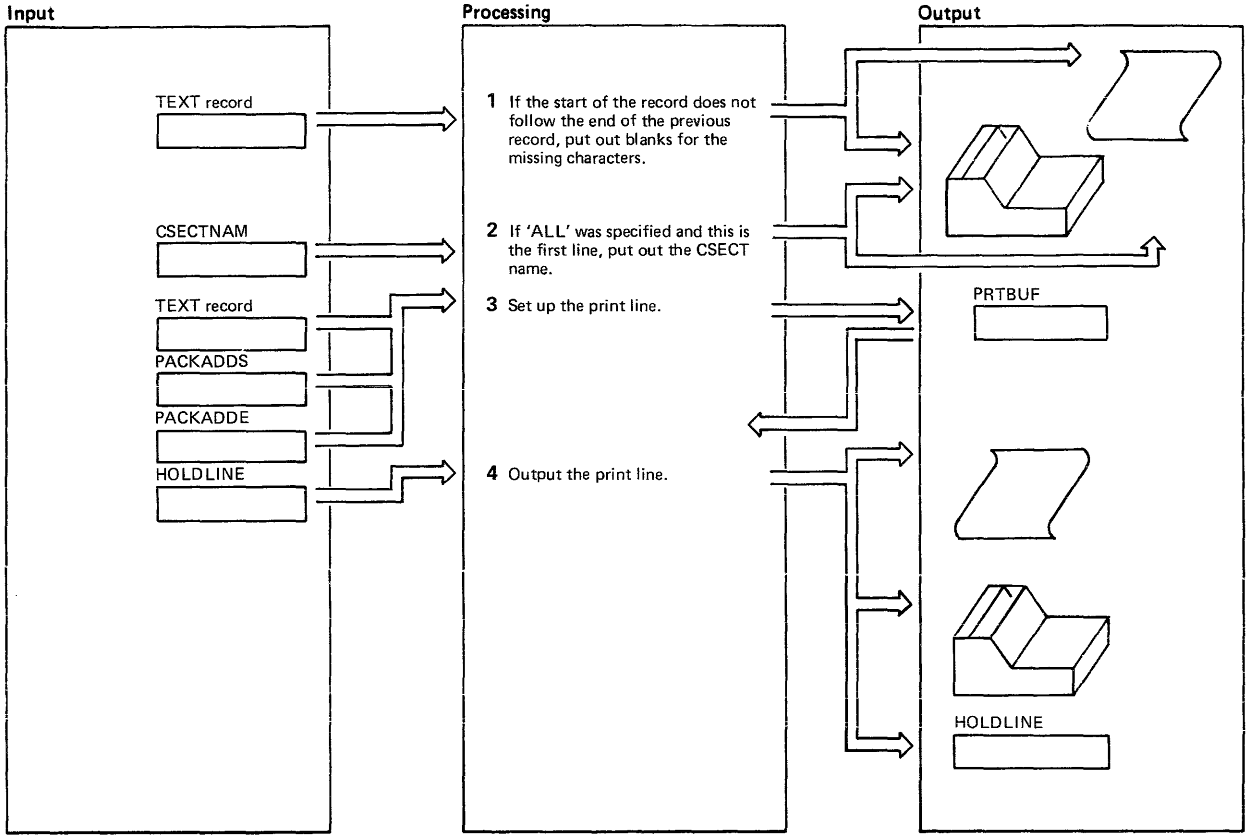
3 Locate the desired position within the record.

4 If DUMP was specified, go to the PRINT routine. (See Diagram 10-9 for details.)

5 Perform the VER or REP operation. If the operation is REP, rewrite the record.

Notes	Moduie	Label	Ref
1 Read the next record of the file. If the file is a module, go to step 3.	DMSZAP	RDTXT	
2 If the file is a TXTLIB, check for the desired record. If not, repeat step 1. Otherwise, check for valid characters. If there are no valid characters (that is, if the area is a Define Storage area), and the operation is VER or REP, issue the message DMSZAP248W INVALID VER/REP DISP – SET NO GO SWITCH	DMSZAP	RDTXTLIB RDTXTFND	
If there are no valid characters, but the operation is DUMP, determine the length of the gap and handle it as a TEXT record.			
If the file is a LOADLIB, check for the desired record. When it is found, check for valid characters as with a TXTLIB and, if valid, read the next record for the actual text.	DMSZAP	RDLDLIB RDLDCHK	
3 Determine the position within the record.	DMSZAP	CHKVER	
4 If the operation desired is DUMP, go to the PRINT routine to print out lines.	DMSZAP	VERCHK PRTDUMP	
5 If the operation is REP, replace each byte read with the data supplied in the REP control record. When the end of the record is reached or the REP operation is completed, rewrite the record.	DMSZAP	VERLOOP VERIFY1 VERIFY2 WRLIB	
If the operation is VER, compare each byte read with the data in the VER control record. If they do not agree, issue the message DMSZAP200W VERIFY REJECT – SET NO GO SWITCH		VERLOOP VERIFY1	
If another record is required, go to step 1. Otherwise, control returns to caller.		RDTXEND	

Diagram 10-8. Reading the Text



1 If the start of the record does not follow the end of the previous record, put out blanks for the missing characters.

2 If 'ALL' was specified and this is the first line, put out the CSECT name.

3 Set up the print line.

4 Output the print line.

Notes	Module	Label	Ref
1 If the start of a new record does not match the end of the previous record, or the requested start of the dump is not found, insert blanks in the output record to represent the bytes not in the file.	DMSZAP	PRTDUMP SETBLANK	
2 If 'ALL' was specified and this is the first line of the CSECT, output the CSECT name.	DMSZAP	NEWLIN PRTHDR	
3 If a line has been started, finish the line. If not, set up the new line. Determine the address of the new line, check that the line does not exceed the requested end of the dump, and move characters from the record into the line. If the line does exceed the requested end of the dump or the record is exhausted, fill the output line as much as possible, convert its characters for printing, and save the pointers. Return control to caller.	DMSZAP	FINLINE NOFSTLN SETADD SETHXLN SETHXLNA CHARCONV PRTRDNXT	
4 When the line is ready for printing, convert the non-printing characters to periods, and compare the line to the previous line. If there is a match, save the address of the current line. If there is no match, and addresses have been saved, print the message LINES xxx TO xxx SAME AS ABOVE. Otherwise, print the line and save it in HOLDINE.	DMSZAP	CHARCONV PRTLINE CHKDUP NOTDUP PRTLIN2	

Notes	Module	Label	Ref

Diagram 10-9. Printing the Dump

Program Organization

This section contains a program description of the DMSZAP module.

DMSZAP

The ZAP service program.

<u>Entry Point</u>
DMSZAP -- via the command ZAP.

<u>Attributes</u>
Reusable, not disk resident.

<u>Entry Conditions</u>
R1: Address of the input parameter list
R15: Address of the entry point

<u>Register Usage</u>
R1: Address of the input parameter list
R2-8: Work registers
R9: Base registers
R10: Link register
R11-12: Base registers
R13: Address of the save area
R14: Return address
R15: Return code

<u>Calls to Other Routines</u>
DMSBRD To read input disk files.
DMSBWR To write output disk files as a result of REP operation.
DMSERR To handle calls from DMSERR and LINEDIT macros.
DMSFNS To close input and output files.
DMSPRT To handle PRINT command.
DMSSMN To handle OS GETMAIN and FREEMAIN macros.
DMSSTT To provide a copy of an FST.
DMSSVT To process OS macros.

<u>External References</u>
None.

<u>Data Areas</u>
File Status Table

<u>Exit Conditions</u>
R15: Return code

Directory

Figure 10-2 is an alphabetical list of the major labels of the ZAP program. The associated method of operation diagrams are indicated and a brief description of the operation performed at the point in the program associated with each label is included.

Label	Diagram	Description
BASEREC	10-4	Processes a BASE control record. Scans for displacement.
CHKLDTBL	10-6	Locates a CSECT (for a module file) if a name is given.
CHKMEM	10-6	Checks for a member, or, if a CMS TXTLIB, for a CSECT.
CLOSELIB	10-6	Finishes the specified library or module.
CLOSINP	10-5	Closes the input file.
CLRSPCE	10-5	FREEMAINs buffer space.
CONEND	10-5	Processes an END control record.
CONSOPT	10-2	Sets the TERM option.
DECODE1	10-4 10-5	Checks that a field is less than six digits.
DECODE2	10-4 10-5	Checks that a field is an even number of digits.
DMSZAP	10-1	Saves the input registers and sets addressability.
DOWTO		Does a write-to-operator for messages when in terminal mode.
DUMPREC	10-3	Gets the location of the dump and prints it.
FDEFINP	10-2	FILEDEFs the input DCB and opens it.
FINDMEM	10-6	Locates the beginning of a member.
FNDCLNTH	10-6	Locates the boundary of a CSECT.
INITOPEN	10-1	Opens input (if specified) and output (printer) files.
INPTOPT	10-2	Sets the INPUT option.
INVEREP	10-2	Processes the error message for an invalid control record and closes the SYSLIB file.
NAMEREC	10-4	Processes a NAME control record. Scans for the member name and CSECT name.
NAMFOUND	10-2	Branches to the appropriate routine when a keyword is found in the table.
NEWLIN	10-9	Prints full lines.

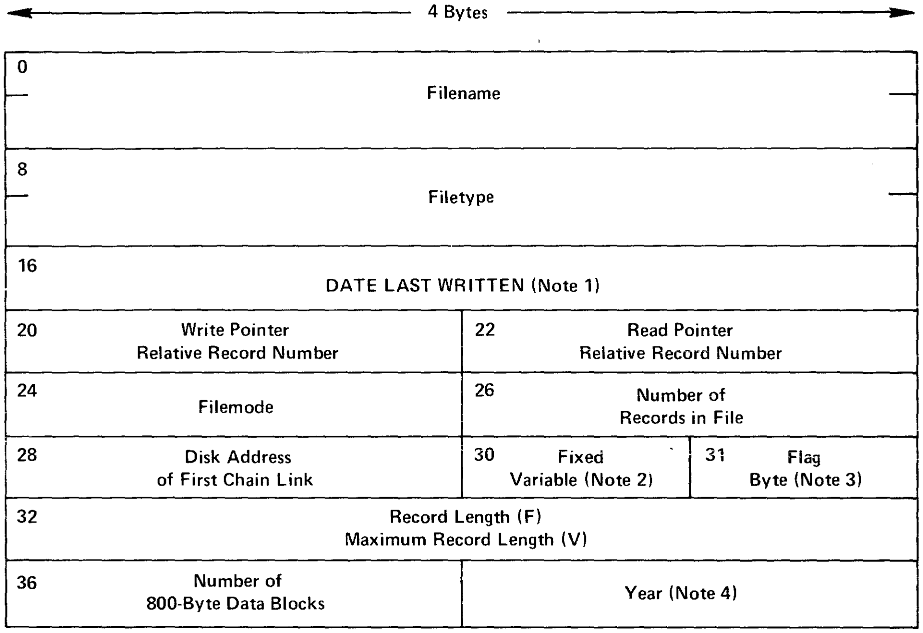
Figure 10-2. The ZAP Program Label Directory (Part 1 of 2)

Label	Diagram	Description
NOMORE	10-5	Gets the error code and prior save area address, restores the registers, and returns to DMSITS.
NOPRTOPT	10-2	Sets the NOPRINT option.
OPENFILE	10-6	Opens a library.
PREOPLB1	10-6	Gets the first library name address.
PREOPLB4	10-6	Reads a ZAP file and locates a member (CSECT for a MODULE file if a name was given).
PREOPLIB	10-6	Opens ZAP files and looks for the library name, if given.
PRINTOPT	10-2	Sets the PRINT option.
PRTCARD		Prints a card image.
PRTDUMP	10-9	Prints the requested dump.
PRTHDR	10-9	Prints the name of the CSECT being dumped.
PRTLINE	10-9	Prints a dump line.
RDCARD	10-2	Requests input from the terminal.
RDCARD2	10-2	Reads an input control record file.
RDLDLIB	10-8	Analyzes LOADLIB records.
RDLIB	10-7	Reads the specified library or module.
RDTXT	10-8	Reads a library searching for the record to be verified or replaced.
RDTXTLIB	10-8	Analyzes TXTLIB records.
READCESD	10-7	Reads a CESD record of a member.
READINP	10-2	Reads a control record from the input file. Writes the control record to the output (SYSPRINT) file. Scans the first keyword from the control record.
SCANKEY1	10-2	Scans control records.
SCANLINE	10-2	Checks the command line for validity.
SEARCHSD	10-7	Searches a CESD record for an ESD entry with a CSECT name.
SETBLANK	10-9	Spaces over a DS area.
STFDEF	10-6	Issues a STATE for a library file, checks that the disk is in Read/Write mode.
TABLOOK	10-2	Look for a keyword in the table.
TXTESD	10-7	Finds a TXTLIB CSECT.
WRCARD	10-2	Writes a control record and messages to SYSPRINT file.
WRLIB	10-8	Updates the specified library or module.

Figure 10-2. The ZAP Program Label Directory (Part 2 of 2)

Data Areas

The File Status Table is used by the DMSZAP module:



4 Bytes		
0 Filename		
8 Filetype		
16 DATE LAST WRITTEN (Note 1)		
20 Write Pointer Relative Record Number	22 Read Pointer Relative Record Number	
24 Filemode	26 Number of Records in File	
28 Disk Address of First Chain Link	30 Fixed Variable (Note 2)	31 Flag Byte (Note 3)
32 Record Length (F) Maximum Record Length (V)		
36 Number of 800-Byte Data Blocks	Year (Note 4)	

Notes:

1. Date last written is in packed decimal format MM DD HH MM; for example, 02 20 14 07 represents February 20, 2:07 p.m.
2. F = Fixed-length records. V = Variable-length records.
3. Flag Byte = 0
4. Year is in character form; for example, '72' for 1972.

Figure 10-3. File Status Table Entry

Diagnostic Aids

THE ZAP COMMAND PROCESSOR (DMSZAP)

Message Code	Label	Diagram	Message Text
DMSZAP001E	SCANLINE	10-2	NO FILENAME SPECIFIED
DMSZAP002W	PREOPLB5	10-6	FILE 'fn ft' NOT FOUND
DMSZAP002E	FDEFINP	10-2	FILE 'fn ft' NOT FOUND
	PREOPLB3	10-6	
DMSZAP003E	SCANLINE	10-2	INVALID OPTION 'option'
DMSZAP007E	FDEFINP	10-6	FILE 'fn ft' IS NOT FIXED, 80 CHAR. RECORDS
	STFDEF		
DMSZAP014E	SCANLINE	10-2	INVALID FUNCTION 'function'
DMSZAP047E	SCANLINE	10-2	NO FUNCTION SPECIFIED
DMSZAP056E	PREOPLB4	10-6	FILE 'fn ft' CONTAINS INVALID RECORD FORMATS
DMSZAP070E	SCANLINE	10-2	INVALID PARAMETER 'param'
DMSZAP104S	PREOPLB4	10-6	ERROR 'nn' READING FILE 'fn ft fm' FROM DISK
	CHKLDTBL	10-6	
	RDCARD2	10-2	
	RDLIB	10-6	
DMSZAP190W	INVEREP	10-2	INVALID CONTROL RECORD OR NO GO SWITCH SET
		10-3	
		10-4	
		10-5	
DMSZAP191W	DUMPREC	10-5	PATCH OVERLAPS - SET NO GO SWITCH
	GOODTHRE		
DMSZAP192W	DECODE1	10-3	ERROR - ODD NUMBER OF DIGITS - SET NO GO SWITCH
		10-5	
	GOODTHRE	10-4	
DMSZAP193W	GOODTHRE	10-5	PRECEDING CONTROL RECORD FLUSHED
DMSZAP194W	OPENFILE	10-6	CSECT NOT FOUND IN '{member|fn} ft' - SET NO GO SWITCH
	READCESD	10-7	
DMSZAP200W	VERIFY1	10-8	VERIFY REJECT - SET NO GO SWITCH
DMSZAP208E	STFDEF	10-6	FILE 'fn ft' IS NOT VARIABLE RECORD FORMAT
DMSZAP210E	STFDEF	10-6	FILE 'fn ft' IS ON A READ/ONLY DISK
DMSZAP213W	BASEREC	10-4	BASE VALID INVALID - SET NO GO SWITCH
DMSZAP245S	WRCARD	10-2	ERROR 'nnn' ON PRINTER
DMSZAP246W	CHKLDTBL	10-6	NO LOADER TABLE PRESENT FOR MODULE 'fn' - SET NO GO SWITCH
DMSZAP247W	PREOPLB3	10-6	MEMBER 'name' NOT FOUND - SET NO GO SWITCH
DMSZAP248W	RDTXTLIB	10-8	INVALID VER/REP DISP - SET NO GO SWITCH
	RDLDLIB	10-7	
DMSZAP750I	CONEND	10-5	ZAP PROCESSING COMPLETE
DMSZAP751I	OPENFILE	10-6	MEMBER FOUND IN LIBRARY 'fn'

Figure 10-4. ZAP Command Processor (DMSZAP) Messages

Chapter 11. DMSIFC and DMSREA—EREP/Error Recording Interface

Introduction

The VM/370 method of editing error records accumulated on the VM/370 error recording cylinders or stored on other devices makes use of the OS/VS EREP Edit and Print programs. To use these programs from a VM/370 virtual machine environment requires the use of the DMSIFC module which is called by DMSITS when the CPEREP (EXEC) command is processed.

DMSIFC loads DMSREA and several modules of OS/VS EREP into main storage and then passes control to OS/VS EREP.

Prior to passing control to EREP, DMSIFC does the following:

- Issues FILEDEFs for files needed by OS/VS EREP.
- Reads control parameters from the user and puts them into an OS-compatible parameter (PARM) list format to be passed to OS/VS EREP.
- Creates a SYSIN file of control parameters from the control parameters that have been entered.
- Uses the HNDSVC macro instruction to prepare for trapping the EXCPs (SVC 0) that OS/VS EREP will issue when it attempts to read records from the SY1.LOGREC data set.
  <u>Note</u>: HNDSVC is also used to prepare to trap BLDLs (SVC 18) that OS/VS EREP will issue.

The several modules of OS/VS EREP that must be loaded by DMSIFC are those that contain VCONs or that are needed in the process of resolving VCONs. DMSIFC invokes the CMS INCLUDE command dynamically to load these OS/VS EREP modules from CPEREP's two TXTLIB files. Other modules of OS/VS EREP that do not contain VCONs are loaded later (from the two TXTLIB files) by OS/VS EREP itself as they are needed.

DMSIFC passes control to OS/VS EREP by executing an OS LINK (to EREP's IFCEREP1 module, which has already been loaded). The OS-compatible parameter list built by DMSIFC is passed to IFCEREP1 at this time and OS/VS EREP begins to execute.

EREP issues set EXCPs for I/O to the OS SYS1.LOGREC data, which are intercepted by CMS. CMS transfers control back to DMSIFC, which simulates the EXCPs so that they appear to access a SYS1.LOGREC data set. This simulation results in calls to DMSREA to supply records contained on the VM/370 error recording cylinders.

EREP issues BLDLs (SVC 18) to determine whether or not EREP modules needed for certain error records are present in the TXTLIBs. The standard CMS simulation of OS BLDL does not include the JOBLIB/STEPLIB form of BLDL which EREP uses here. Therefore, these BLDLs are intercepted and are simulated by DMSIFC.

When EREP is finished executing, it exits (returns to DMSIFC which invoked it). Before returning to CMS, DMSIFC does some cleaning up. Temporary files are erased and FILEDEFs issued by DMSIFC are cleared with the following exceptions: the EREPPT, ACCIN, and ACCDEV FILEDEFs are not cleared because they may have been entered by the user or by DMSIFC but DMSIFC has no way of knowing which. Since they should not be cleared if they were entered by the user, DMSIFC never clears them.

In order to make use of the CPEREP command, both of the following publications are required. The first publication provides general information on the use of the command and detailed information on command operands applicable only to VM/370. The second publication provides detailed information on the operands that are common to both VM/370 and OS/VS.

> <u>IBM Virtual Machine Facility/370</u>: <u>OLTSEP and Error Recording Guide</u>, Order No. GC20-1809.
>
> <u>OS/VS Environment Recording Editing and Printing (EREP) Program</u>, Order No. GC28-0772.

Program logic information describing OS/VS EREP is contained in:

> <u>OS/VS Environment Recording Editing and Printing (EREP) Program Logic</u>, Order No. SY28-0773.

Method of Operation

This section describes the VM/370 interface between CMS (the Conversational Monitor System) and the OS/VS EREP program. Diagrams 11-1 and 11-2 describe the functions of the interface modules and serve as a guide to the program listings. The labels shown indicate the closest, nonmacro expansion label to the function being documented. These diagrams are not terribly detailed, therefore, some functions are not shown. Use the Directory and Program Organization section to find the labels in the program listings for any routines that are not shown in the Method of Operation section. Figure 11-1 shows the relationship of these diagrams.

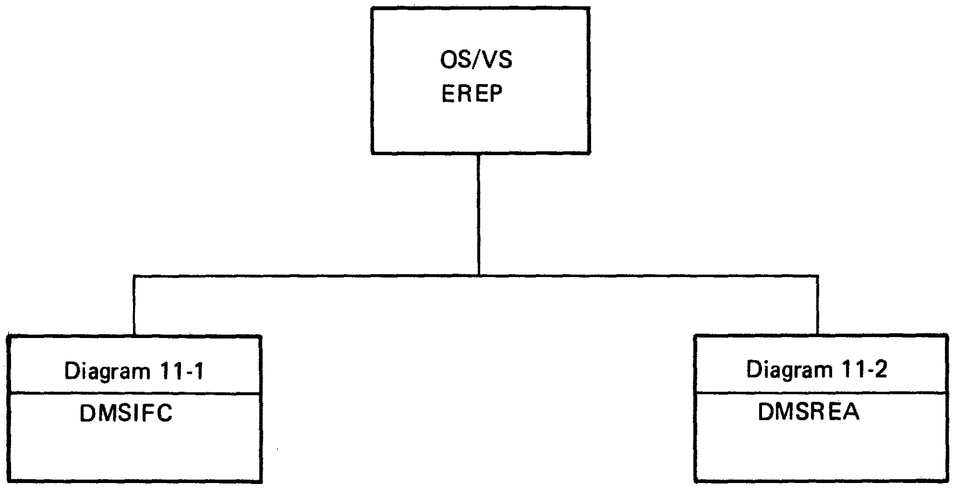
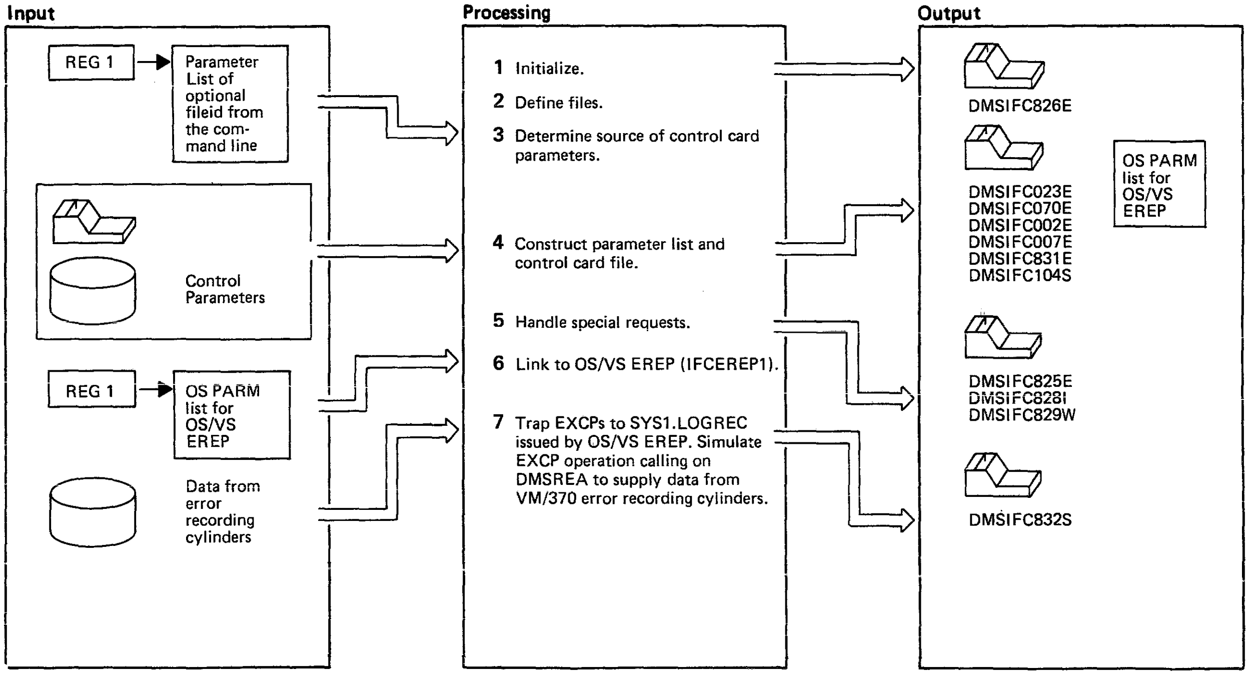


Figure 11-1. Key to EREP/Error Recording Interface Method of Operation Diagrams



1 Initialize.

2 Define files.

3 Determine source of control card parameters.

4 Construct parameter list and control card file.

5 Handle special requests.

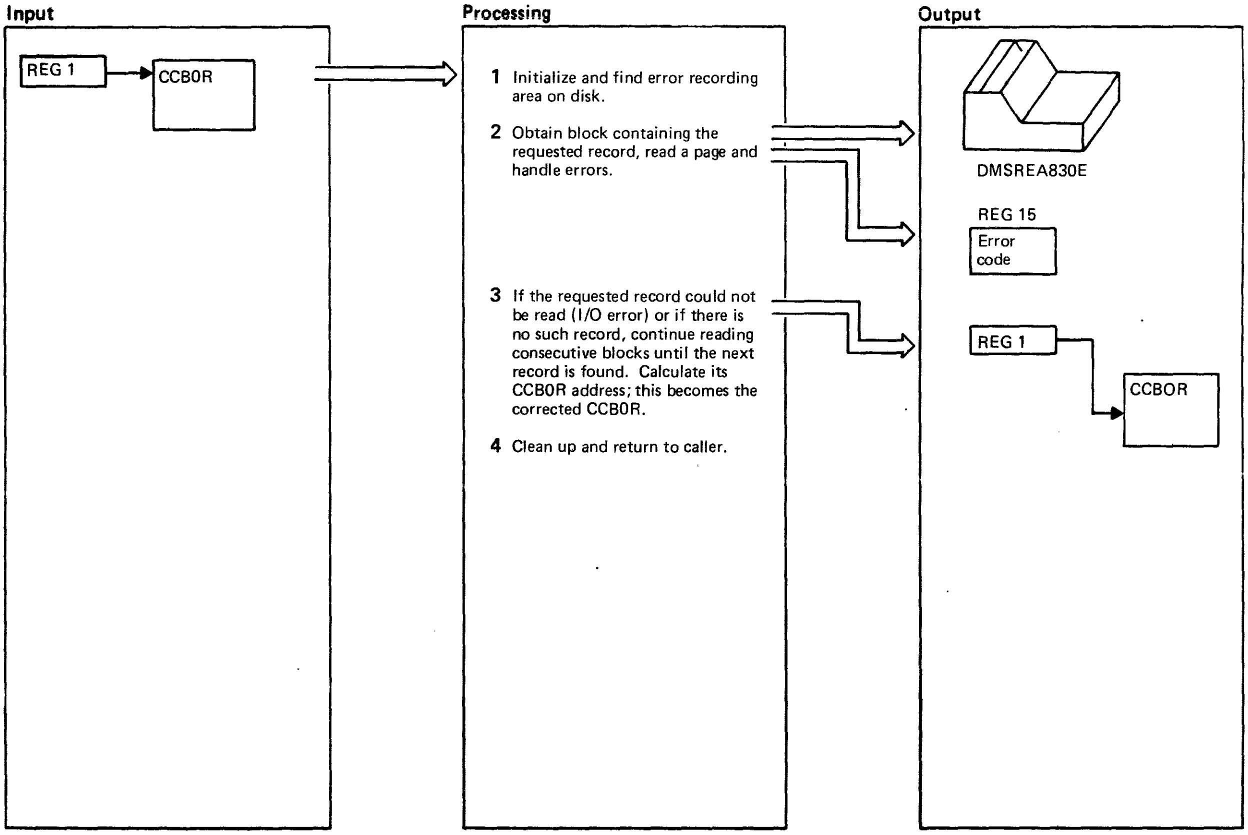
6 Link to OS/VS EREP (IFCEREP1).

7 Trap EXCPs to SYS1.LOGREC issued by OS/VS EREP. Simulate EXCP operation calling on DMSREA to supply data from VM/370 error recording cylinders.

Notes	Module	Label	Ref
1 The initialization procedures include: ● Standard linkage and addressability functions. ● Loading and resolving VCONs in OS/VS EREP decks. ● Loading DMSREA. ● Turning off flag in CMS nucleus to cause OS simulation. ● Setting COMPSWT in CMS nucleus to load LINK and LOAD macros to be entered in TEXT files. ● Establishing handling of SVC 76, SVC 18, and SVC 0.	DMSIFC	DMSIFC	Diagram 11-1
2 Invoke FILEDEF to define:			
● Printer file (EREPPT).		NORWDISK	
● SYSIN file (SYSIN).		RDYACC	
● Dummy file for SYS1.LOGREC (SERLOG).		RDYHIST	
● Error file (TOURIST).		OPER12	
● Work file (DIRECTWK). ● Accumulation tape file (ACCDEV). ● History input tape (ACCIN).			
3 Determine where control parameters are to be taken from (Control file or terminal).		HAVETYPE NOEXTRA BADATTR GOODATTR	
4 Set up to read parameters. Obtain storage for OS PARM list to be passed to EREP. Read control parameters, generating the OS PARM list and a SYSIN file as output. Call subroutine to read control parameters. Handle errors.		PARMWORK RDERR1 PLISTBLD	Diagram 11-1
5 If CLEAR is specified with other parameters, type an error message. If CLEAR is specified properly, call subroutine to erase error records from the VM/370 error recording cylinders. Subroutines handle each parameter information:		WANTCLR CLEARRTN	

Notes	Module	Label	Ref
CLEARF parameter (determines validity by examining processor identity. If not 3031, 3032, or 3033 processor reject command but if valid, erase error records from the error recording cylinders then initialize SRF frames to the beginning of the error recording cylinders.)		HCLEARF	
● CLEAR parameter.		HCLEAR	
● TERMINAL parameter (stops reading from control file on disk and goes to terminal to read additional control parameters).		HTERM	
● SHARE parameter.		HSHARE	
● CTLCRD parameter.		HCTLCRD	
● ACC parameter.		HACC	
● HIST parameter.		HHIST	
● MERGE parameter.		HMERGE	
● MES and THRESHOLD parameters.		HMES	
● RDESUM parameter.		HRDESUM	
● ZERO parameter.		HZERO	
6 Load the address of the word that points to the OS PARM list built for OS/VS EREP and LINK to IFCEREP1.			
7 EXCP SVCs from EREP are intercepted and simulated so they appear to access a SYS1.LOGREC data set. Simulation causes calls to DMSREA for VM/370 error records. BLDL SVCs from EREP are also trapped and simulated by DMSIFC.		DMSIFC0 DMSIFC18	

Diagram 11-1. DMSIFC



1 Initialize and find error recording area on disk.

2 Obtain block containing the requested record, read a page and handle errors.

3 If the requested record could not be read (I/O error) or if there is no such record, continue reading consecutive blocks until the next record is found. Calculate its CCBOR address; this becomes the corrected CCBOR.

4 Clean up and return to caller.

Notes	Module	Label	Ref
1 The initialization procedures include: ● Saving registers. ● Setting return code to zero. ● Issuing DIAGNOSE X'2C' to locate beginning of error recording area and number of cylinders.	DMSREA	DMSREA	Diagram 11-2
● Setting and checking "first time" switch.		FIRSTSW	
● Checking CCBOR address passed for validity. Note: A CCBOR disk address is a disk addressing format devised solely for use in CPEREP and resembles the commonly used CCHHR disk address. In a CCBOR address the fields have the following meaning: Field Meaning CC Relative cylinder within the VM/370 error recording area, for example: CC = X'0000' for the first cylinder of the error recording area. = X'0001' for the second cylinder of the error recording area. B The number of the desired 4K block within the cylinder. The first 4K block in a cylinder is X'01'. 0 Zero. R The number of the desired record within the 4K block. The first record in a block is X'01'.		OPER4	

Notes	Module	Label	Ref
2 DMSREA converts the CCBOR address to a VM/370 Control Program Internal Format address and issues a DIAGNOSE X'30' to read the block into the buffer. If the requested block is found, return to caller. If specified cylinder is outside error recording area, sets error code in register 15 for invalid cylinder. If end of cylinder and no more cylinders are available, sets register 0 to zero, indicating end-of-file to caller; otherwise, advance to next cylinder. If an I/O error occurs so that the block could not be read, issue message DMSREA830E.		OPER5 OPER7 OPER16 OPER17 OPER7 OPER9	
3 If requested record was not found, read next block and return first record from this block. If block is empty or unreadable, continue reading blocks until a record is found or until end-of-file is reached. Use CCBOR address of the record found as the corrected CCBOR value to be returned to the caller. Make register 1 point to this CCBOR address. Note: The CCBOR record addresses are passed back to OS/VS EREP (as a result of the EXCP simulation) as if they were CCHHR addresses. EREP never notices the difference and, as a result, EREP uses CCBOR addresses in all its I/O operations to the SYS1.LOGREC data set.		OPER10	
4 Restore registers (except output parameter registers) and return to caller.		OPER15	

Diagram 11-2. DMSREA

Program Organization

This secticn includes program descriptions of modules DMSIFC and DMSREA.

DMSIFC

Allows virtual users to edit and print VM/370 error recordings under CMS via the OS/VS EREP Edit and Print Program (IFCEREP1).

Entry Point
DMSIFC

Routines Called
IFCEREP1 via LINK to edit and print VM/370 error recording cylinders.
DMSREA via BALR to read a specified record from the VM/370 error recording cylinders.
DMSLAD via BALR to determine which read/write disk has the most space.
DMKIOG via DIAGNOSE to clear requested recording cylinders.
STATE/STATEW via SVC to perform CMS functions.
ERASE via SVC to perform CMS functions.
INCLUDE via SVC to perform CMS functions.

Attributes
Nonreusable, CMS User Area, and called by CMS.

Registers at Entry
R1: CMS parameter list address
R13: Save area address
R14: Return address

Registers at Exit
R0-R14: Restored
R15: One of the following return codes

Return Code	Meaning
12	CLEAR specified with other parameters.
24	An invalid parameter or no filetype was specified.
28	The file was not found.
32	The file was not a fixed-length format.
56	GLOBAL command was not issued for CPEREP's TXTLIBs.
60	An I/O error caused one or more of the 4K blocks of error records to be skipped.
62	More than the maximum number of characters in options specified.
88	Attempt tc set to zerc was suppressed. Requires privilege class F.
100	Error reading file frcm disk.

Register Usage

R0-R1:	Parameter registers
R2-R9:	Scratch
R10-R11:	Spares, not used
R12:	Base register
R14-R15:	Link registers

External References
CURRSAVE Contains address of the current system save area when control is received to handle an SVC as requested by the HNDSVC macro.
OSSFLAGS OS simulation flags in the NUCON area.
DOSFLAGS DOS simulation flags in the NUCON area.
AADTLKW Contains address of rcutine that determines which read/write disk has the mcst space. (In the NUCON.)
TXTLIBS Indicates whether or nct any TXTLIBs have been globaled. (In the NUCCN.)
TXTDIRC Indicates whether or nct any TXTLIBs have been globaled. (In the NUCCN; points to the first directory in the chain of global TXTLIB directories.)

The functions performed by DMSIFC can be summarized as follows:

1. Performs standard linkage and addressability functicns.

2. Invokes CMS LOAD function tc load and resolve VCONs in about a dczen EREP object decks.

   Note: All other EREP object decks are brought into stcrage later, as needed, by OS LOAD and LINK macrcs issued by OS/VS EREP.

3. Invokes STRINIT function. Indicates that area abcve presently loaded programs is the beginning cf free storage.

4. Turns off the DOSSVC flag in the CMS nucleus so that OS simulaticn can be used. Sets CCMPSWT in CMS nucleus so that OS LCAD and LINK macros bring in TEXT files rather than module files. Invokes OS LOAD

to load DMSREA into storage and saves its address so it can be called later during the EXCP simulation.

5. Establishes handling of SVC 76 (error log), SVC 18 (BLDL), and SVC 0 (EXCP).

6. Invokes FILEDEF function to define:
   - Printer file for EREP
   - SYSIN file to be created for EREP
   - Dummy file for EREP to open and close as SYS1.LOGREC
   - "TOURIST" error file to the terminal
   - DIRECTWK work file on disk

7. Gets the command line arguments and determines if a control file is provided for input. If so, sets up to read parameters from the control file, otherwise, sets up to read parameters from the terminal.

8. Issues a DMSFREE macro to get storage for building OS parameter list to be passed to EREP.

9. Gets input parameters (from control file or terminal) and constructs equivalent OS/VS EREP parameter list and SYSIN control card file.

10. If CLEAR was specified, and it was not the only parameter specified, types an error message to the terminal and does housekeeping and exits to CMS.

11. If CLEAR was specified correctly, calls a subroutine to issue the DIAGNOSE that clears the appropriate records from the VM/370 error cylinders, then does housekeeping and exits to CMS. If CLEARF was specified, read CPU and director frames from SRF device and write on error cylinder.

12. Invokes FILEDEF to define the accumulation tape file if requested. Issues the tape control macros necessary to position tape for subsequent write operations.

13. Invokes FILEDEF to define history input tape if requested and makes sure that it is rewound.

14. Links to OS/VS EREP (IFCEREP1).

15. Simulates BLDL SVCs issued from OS/VS EREP. Simulates EXEC SVCs issued from OS/VS EREP so they will appear to access a SYS1.LOGREC data set. EXCP simulation will result in calls to DMSREA to get records from VM/370 error recording cylinders. An EXCP that attempts to rewrite the SYS1.LOGREC header is a result of the ZERO function and is simulated by calling a subroutine to issue the DIAGNOSE that clears the error records from the error recording cylinders.

16. Eventually OS/VS EREP is done and control returns from that LINK done above.

17. Housekeeps all indicators and switches, frees any storage obtained for the OS parameter list area, clears handling of SVC 0, SVC 18, and SVC 76; and clears any FILEDEFs that were set up by CPEREP.

18. Exits to CMS.

DMSREA

Reads a specified logical record from the VM/370 error recording cylinders and returns it to the caller.

<u>Entry Point</u>
DMSREA

<u>Routines Called</u>
DIAGNOSE X'2C' to find the beginning of the recording area on the system disk, and the number of error recording cylinders.
DIAGNOSE X'30' to read a page size record from the error recording cylinders.
DMSERR via macro SVC to write error messages to the console.

<u>Attributes</u>
Nonreusable, CMS User Area, enter via CALL.

<u>Registers at Entry</u>
R1: Address of CCBOR DASD record address
R13: Save area address
R14: Return address

<u>Register at Exit</u>
R0: Nonzero: address of variable-length record being returned.
The first 4 bytes are the record descriptor word containing the record length.
Zero: end-of-file; no record was at or beyond the entered address.
R1: Address of CCBOR DASD record address (sometimes corrected).
R13: Save area address.

R15: One of the following return codes:

Return Code	Meaning
00	Nothing unusual.
04	Empty 4K block skipped.
08	Invalid CC value in CCBOR address that was entered.
60	I/O error accompanied by message DMSIFC830E.

Register Usage

R0-R9:	Scratch
R10-R11:	Spares, not used
R12:	Base
R13:	Save area address
R14-R15:	Scratch

External References

None.

The functions performed by DMSREA can be summarized as follows:

1. Issues the DIAGNOSE command to find the beginning of the VM/370 error recording cylinders and the number of cylinders.
2. Reads a requested record from the VM/370 error recording cylinders.
3. Returns the next logical record to the caller when the requested record does not exist or cannot be read and revises the caller's specified CCBOR address accordingly.
4. Handles errors.

Directory

Figure 11-2 is an alphabetical list of the major labels cf modules DMSIFC and DMSREA. The associated method of operation diagrams are indicated and a brief description of the operation performed at the point in the program associated with each label is included.

Label	Diagram	Description
BADATTR	11-1	Handles file not fixed.
CLEARRTN	11-1	Logically erases VM/370 error recording cylinders.
DMSIFC0	11-1	Handles trapped EXCPs issued by EREP.
DMSIFC18	11-1	Handles trapped OS BLDL macros issued by EREP.
OPER7	11-2	Issues I/O error reading records message.
OPER9XX	11-1	Handles specification of CLEAR when entered with other parameters.
NOEXTRA	11-1	Handles file not found.
EXIT0	11-1	Restores registers for exit from DMSIFC.
EXIT1	11-1	Clears handling of SVCs.
EXIT3	11-1	Frees storage allocated for OS parameter list.
EXIT9	11-1	Frees storage allocated for SVC simulation.
FIRSTSW	11-2	Sets indication of first time DMSREA is called.
HACC	11-1	Directs addition of ACC parameter to OS parameter list being built for EREP.
HAVETYPE	11-1	Handles the specification of an extra parameter on the CPEREP command line.
HCLEAR	11-1	Clears all error records from the error recording cylinders.
HCLEARF	11-1	Clears SRF frame records and all error records and reformats the error recording cylinders.
HCTLCRD	11-1	Writes CTLCRD information into SYSIN file for EREP to read.
HHIST	11-1	Directs addition of HIST parameter to OS parameter list being built for EREP.
HMERGE	11-1	Directs addition of MERGE parameter to OS parameter list being built for EREP.
HMES	11-1	Directs addition of MES and THRESHOLD parameters to OS parameter list being built for EREP.
HRDESUM	11-1	Directs addition of RDESUM parameter to OS parameter list being built for EREP.

Figure 11-2. DMSIFC and DMSREA label Directory (Part 1 of 2)

Label	Diagram	Description
HSHARE	11-1	Writes SHARE parameter into SYSIN file for EREP to read.
HZERO	11-1	Directs addition of ZERO parameter to OS parameter list being built for EREP.
OPER4	11-2	Checks CC portion of entered CCBOR for valid range.
OPER7	11-2	Prepares for and issues DIAGNOSE command to read a page of error records.
OPER9	11-2	Prepares to read first record of next block.
OPER10	11-2	Retains address of block just read into buffer. Decides whether this block contains data or is empty.
OPER12	11-1	Handles special considerations for ACC parameter specification.
OPER13	11-1	Handles special considerations for HIST parameter specification.
OPER15	11-2	Restores registers and returns to caller from DMSREA.
OPER16	11-2	Sets error code for invlaid cylinder.
OPER17	11-2	Handles end of cylinder indication.
PARMWORK	11-1	Issues DMSFREE macro to get storage for building OS parameter list.
PLISTBLD	11-1	Adds passed parameters to OS parameter list being built for EREP.
RECLOOP	11-1	Increments counters to step through buffer until empty or end of specified record found.
RDCTLINE	11-1	Reads and returns one line of control parameters from the terminal or control file.
RDERR1	11-1	Handles errors reading control file from disk.
WANTCLR	11-1	Handles calling subroutine to perform CLEAR.

Figure 11-2. DMSIFC and DMSREA Label Directory (Part 2 of 2)

Data Areas

DMSREA

No system data areas are used by DMSREA. However, DMSREA uses 4K of unallocated storage at absolute location X'21000' as a page buffer in which to read the 4K blocks of error records.

DMSIFC

DMSIFC uses ADTECT (the ADT macro) and FSTSECT (FSTB macro) to read from but does not store into them. It uses SSAVE and NUCON also. SSAVE is the CMS system save area that saves the value of the SVC old PSW, the caller's registers, and other necessary control information required to process SVCs and return to the caller. NUCON contains all the nucleus constants for CMS. These are either listed at the end of the module or a description can be found in the _VM/370 Data Areas and Control Block Logic_ manual.

Diagnostic Aids

Figure 11-3 lists the messages issued by DMSIFC and DMSREA. The label of the message and the associated method of operation diagram in which it is documented are included in the list.

Message Code	Label	Diagram	Message Test
DMSIFC002E	NOEXTRA	11-1	FILE 'fn ft [fm] ' NOT FOUND
DMSIFC007E	BADATTR	11-1	FILE 'fn ft fm' IS NOT FIXED. 80 CHAR. RECORDS
DMSIFC023E	NORWDISK	11-1	NO FILETYPE SPECIFIED
DMSIFC070E	HAVETYPE	11-1	INVALID PARAMETER 'parameter'
DMSIFC104S	RDERR1	11-1	ERROR 'nn' READING FILE 'fn ft fm' FROM DISK
DMSIFC825E	OPER9XX	11-1	'CLEAR' IS VALID ONLY WHEN SPECIFIED BY ITSELF
DMSIFC826E	DMSIFC	11-1	EREP TXTLIBS NOT FOUND
DMSIFC828I	CLROKAY	11-1	CPEREP ZERO OR CLEAR HAS BEEN COMPLETED
DMSIFC829W	CLEARRTN	11-1	ATTEMPTED 'ZERO' WAS SUPPRESSED. REQUIRES PRIVILEGE CLASS F
DMSIFC831E	PLISTBLD	11-1	MORE THAN 100 CHARS. OF OPTIONS SPECIFIED
DMSIFC832S	EXGENERR	11-1	SOFTWARE INCOMPATIBILITY AT THE CPEREP-EREP INTERFACE. CODE=nnn
DMSREA830E	OPER7	11-2	I/O ERROR READING A BLOCK OF RECORDS FROM THE ERROR RECORDING CYLINDERS

Figure 11-3. DMSIFC and DMSREA Messages

Chapter 12. DMKMSS—The MSS Communicator

Introduction

The DMKMSS program operates under the control of either OS/VS1 or OS/VS2 (MVS) in a virtual machine. It is a communications interface between the VM/370 control program and the MSS Mass Storage Control. It uses a combination of CP-generated attention interrupts on a virtual I/O device, the DIAGNOSE code X'78' instruction, and OS/VS SVC 126 to provide communications.

Requests are received from VM/370 in response to a DIAGNOSE code X'78' instruction issued by DMKMSS. They are passed to the MSC using the standard OS/VS SVC 126. Responses are received from the MSC and returned to VM/370 using DIAGNOSE.

Method of Operation

This section describes the two major sections of the DMKMSS program.

Diagram 12-1 shows initialization using OS/VS control blocks.

Diagram 12-2 shows the processing of a VM/370 request.

Figure 12-1 shows the relationship of these diagrams.

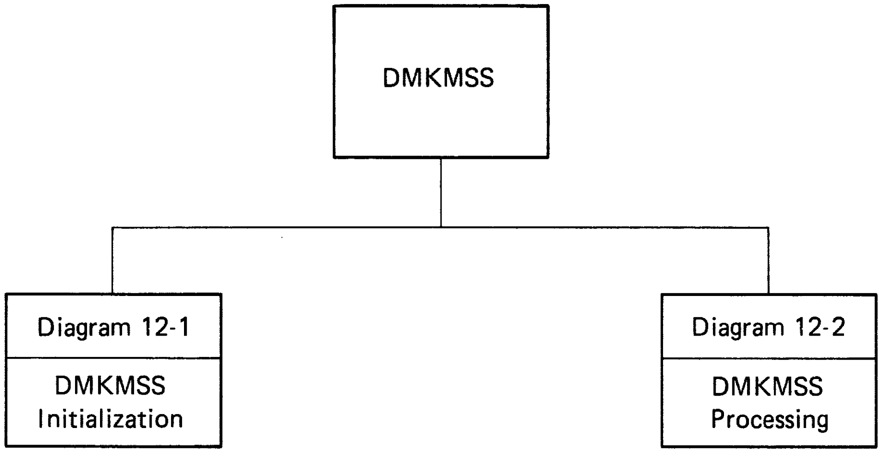
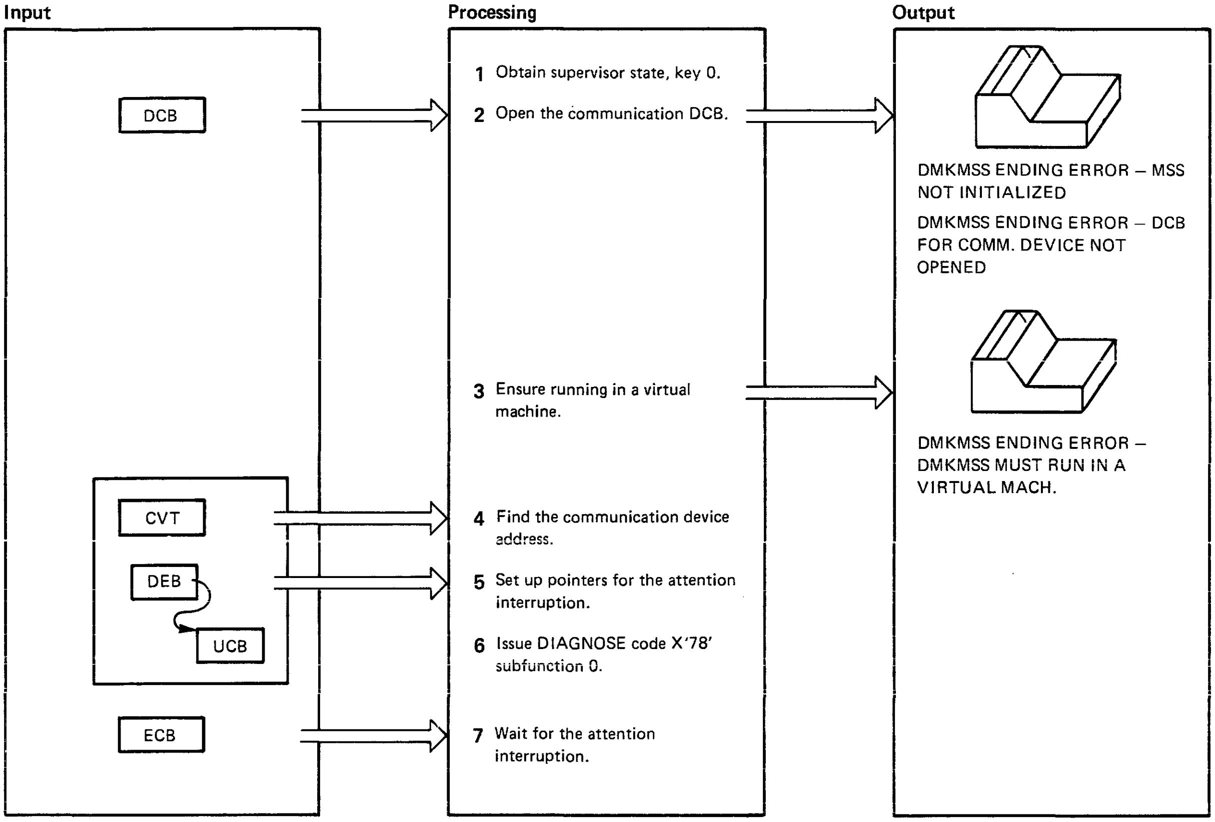


Figure 12-1. Key to the DMKMSS Method of Operation Diagrams



1 Obtain supervisor state, key 0.

2 Open the communication DCB.

3 Ensure running in a virtual machine.

4 Find the communication device address.

5 Set up pointers for the attention interruption.

6 Issue DIAGNOSE code X'78' subfunction 0.

7 Wait for the attention interruption.

DMKMSS ENDING ERROR – MSS NOT INITIALIZED

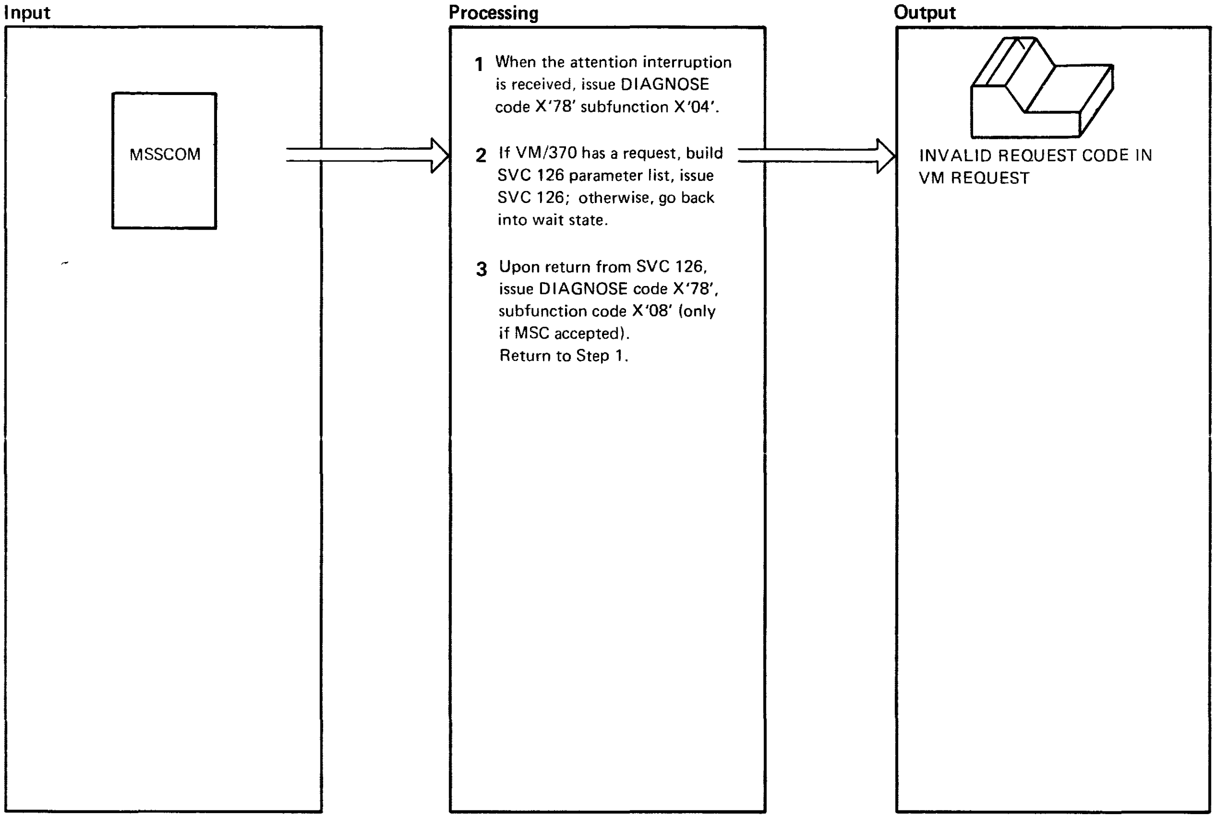
DMKMSS ENDING ERROR – DCB FOR COMM. DEVICE NOT OPENED

DMKMSS ENDING ERROR – DMKMSS MUST RUN IN A VIRTUAL MACH.

Notes	Module	Label	Ref
1 Use the VS MODESET SVC to get into supervisor state, key 0.	DMKMSS		
2 Use the VS OPEN SVC to connect the DCB to the VS control block. If MSS initializes incorrectly, issue message. If the DCB for the communication device does not open, issue message.		RF00092 RF00182	
3 Use the STIDP instruction to ensure running in a virtual machine. If not running in a virtual machine, issue message.		RF00082 RF00190	
4 Follow pointers through the DCB, DEB, and UCB control blocks to find the channel/unit address assigned by the VS scheduler.		L1	
5 Set the MSC's attention table index in the communication device's UCB. Also store the address of the ECB to be waited on in an unused field of this same communication UCB.			

Notes	Module	Label	Ref
6 Build and issue the DIAGNOSE code X'78' instruction to tell VM/370 the channel/unit address.		PROLOG	
7 Issue VS WAIT SVC, specifying that the event control block will be posted when the attention interruption is received.			

Diagram 12-1. DMKMSS Initialization



1 When the attention interruption is received, issue DIAGNOSE code X'78' subfunction X'04'.

2 If VM/370 has a request, build SVC 126 parameter list, issue SVC 126; otherwise, go back into wait state.

3 Upon return from SVC 126, issue DIAGNOSE code X'78', subfunction code X'08' (only if MSC accepted).
Return to Step 1.

Notes	Module	Label	Ref
1 This loop will run in the VS machine as long as MSS support is in effect. The DIAGNOSE X'78' instruction points to a buffer in DMKMSS into which VM/370 places an MSSCOM, or zeros.	DMKMSS	MAINLOOP	
2 Look at MSSCOM to determine volume serial, 3330V device address, and type of request (mount or demount). If the request is invalid, issue a message. If there are no outstanding requests, go into a wait state.		L2 RF00149 RF00122	
3 The SVC 126 routines issue orders to the MSC. If the MSC rejects the order, it sends a unit check as ending status. SVC then sets a non-zero return code in register 15.		DIAG MSSCHECK	

Notes	Module	Label	Ref

Diagram 12-2. DMKMSS Processing

Program Organization

This section describes the program organization of the DMKMSS module.

DMKMSS

The MSS communicator program.

Attributes
Reentrant

Entry Point
DMKMSS

Register Usage

R0-R9:	Work registers
R10:	Workarea base
R11:	Program base
R12:	Work register
R13:	Register savearea base
R14-R15:	Work registers

Directory

Figure 12-2 is an alphabetical list of the major labels in the DMKMSS program. The figure indicates the associated method of operation diagrams and it provides a brief description of the operaticn performed at the point in the program associated with each label.

Label	Diagram	Description
DIAG	12-2	Issues DIAGNOSE code X '78' sutfunction X'08' or X'0C'.
L1	12-1	Follows pointers through the ECB, DEB, and UCE to find the communicator device address.
L2	12-2	Determines the type of MSS request (mount or demcunt).
MAINLOOP	12-2	Issues DIAGNOSE code X'78' subfunction X'04', requesting wcrk.
MSSCHECK	12-2	Sets the MSC completion code for VM/370.
PROLOG	12-1	Initializes for DIAGNOSE code X'78' sutfunction X'00'.
RF00082	12-1	Issues STIDP instruction to ensure running in a virtual machine.
RF00092	12-1	Issues message that MSS is not initialized.
RF00122	12-2	Waits for the communicator device attention interruption.
RF00149	12-2	Issues message for invalid request code in VM request.
RF00182	12-1	Issues message that DCE is not opened.
RF00190	12-1	Issues message that this must run in a virtual machine.

Figure 12-2. DMKMSS Label Directory

Data Areas

The OS/VS control blocks used (CVT, DCB, DEB, and UCB are described in OS/VS1 System Data Areas, Order No. SY28-0605, and in OS/VS2 System Debugging Library: Debugging Handbook, Order No. GC28-0632.

The MSS communicator control block (MSSCOM) is described in IBM Virtual Machine Facility/370 Data Areas and Control Block Logic, Order No. SY20-0884.

Diagnostic Aids

Figure 12-3 lists the messages issued by the DMKMSS program. The nearest label and the associated method of operation diagram are identified.

Label	Diagram	Message Text
RF00092	12-1	DMKMSS ENDING ERROR - MSS NOT INITIALIZED
RF00149	12-2	INVALID REQUEST CODE IN VM REQUEST
RF00182	12-1	DMKMSS ENDING ERROR - DCB FOR COMM. DEVICE NOT OPENED
RF00190	12-1	DMKMSS ENDING ERROR - DMKMSS MUST RUN IN A VIRTUAL MACH.

Figure 12-3. DMKMSS Messages

Chapter 13. DMKIMG and DMKNMT—IEBIMAGE Interface

Introduction

The GENIMAGE and IMAGELIB utility programs enable the installation to dynamically change the character arrangement tables, graphic modifications, copy modifications, and FCBs for the 3800 Print Subsystem.

The first program (module DMKIMG) invoked by the GENIMAGE CMS command, creates TEXT files on a CMS disk. These TEXT files are the images that will be used by the 3800. The IMAGELIB program (module DMKNMT) invoked by the IMAGELIB command, loads the necessary TEXT decks into the named system allocated at system generation time.

Method of Operation

This section describes the VM/370 interface between CMS and the IEBIMAGE program. Diagrams 13-1 and 13-2 describe the functions of the interface modules and serve as a guide to the program listings. The labels shown indicate the closest label to the function being documented. Use the Directory and Program Organization sections to find the labels in the program listings for any routines that are not shown in the Method of Operation section. Figure 13-1 shows the relationship of these diagrams.

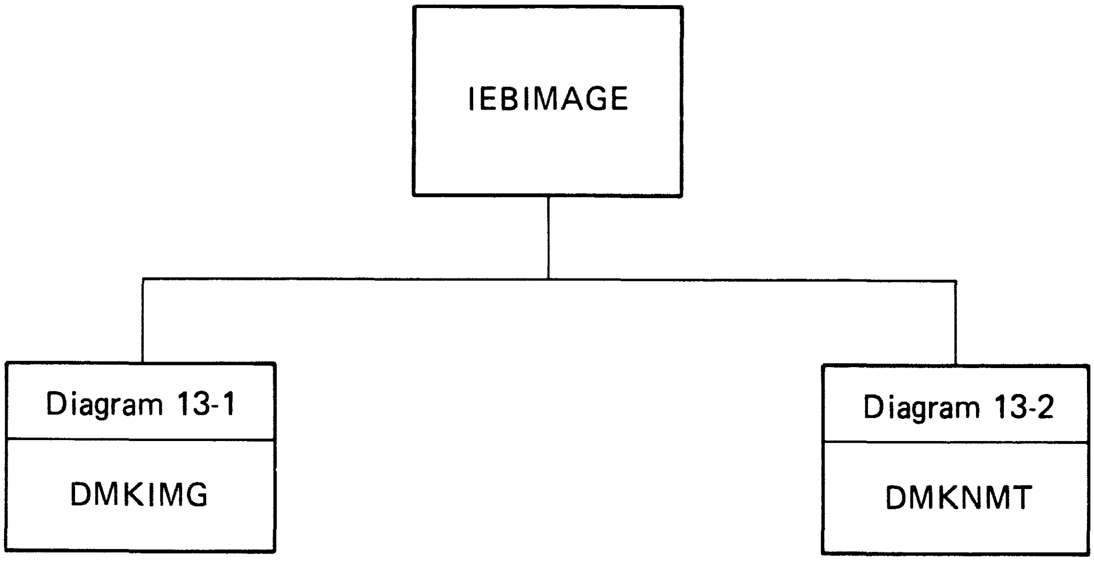
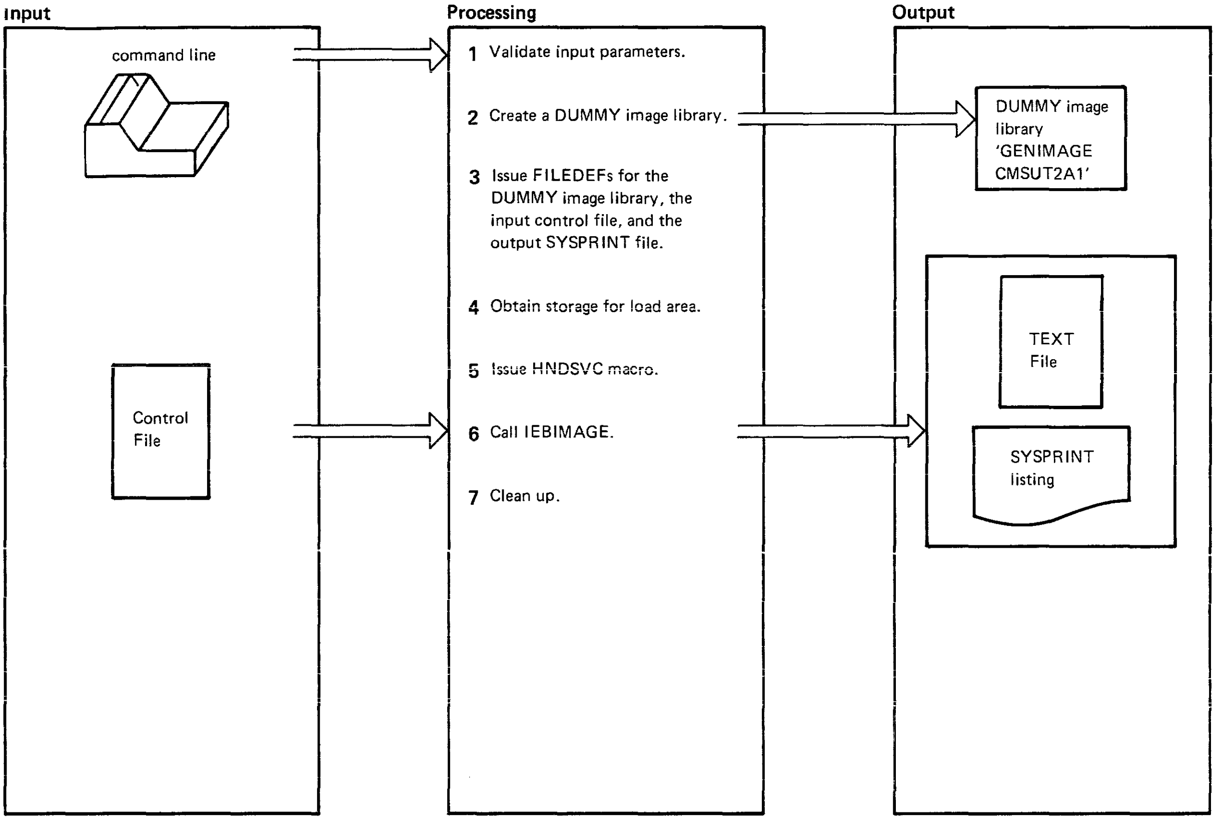


Figure 13-1. Key to the IEBIMAGE Interface Method of Operation Diagrams



1 Validate input parameters.

2 Create a DUMMY image library.

3 Issue FILEDEFs for the DUMMY image library, the input control file, and the output SYSPRINT file.

4 Obtain storage for load area.

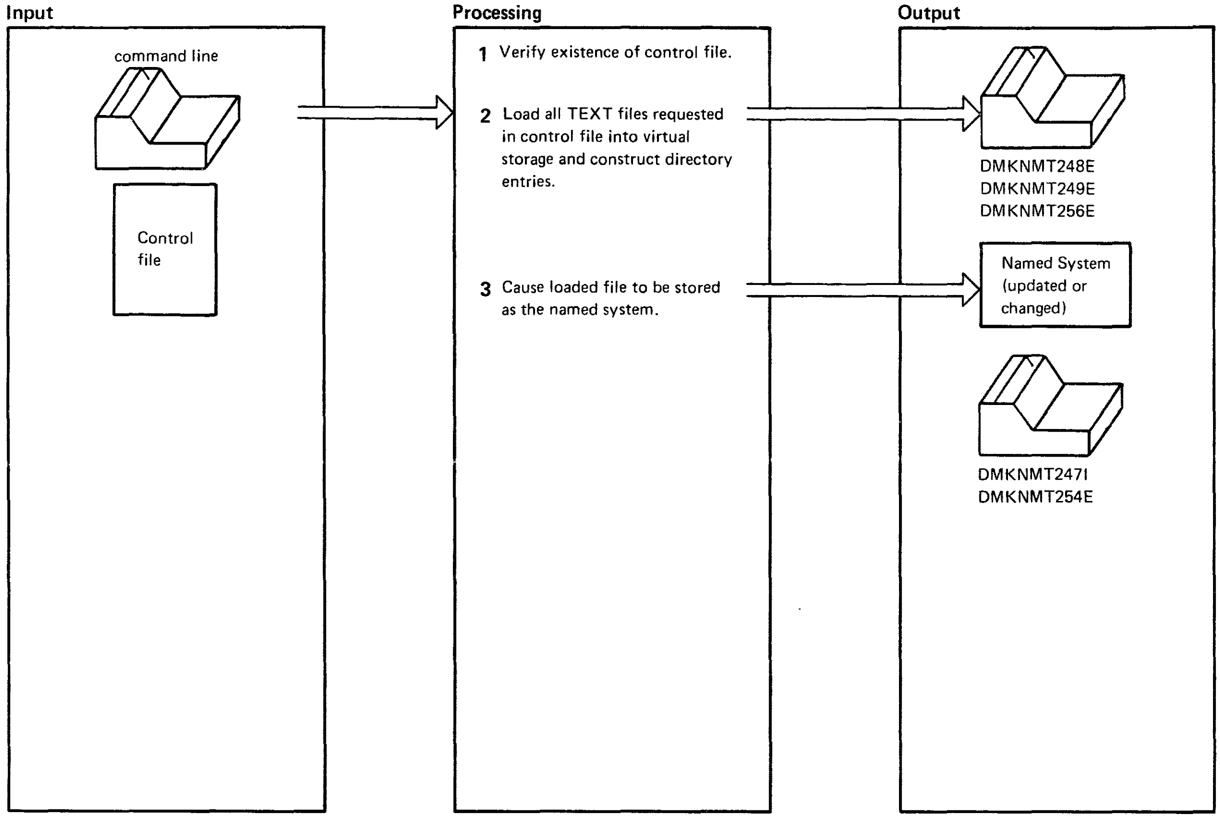
5 Issue HNDSVC macro.

6 Call IEBIMAGE.

7 Clean up.

Notes	Module	Label	Ref
1 GENIMAGE command parameters are validated. If a parameter is invalid, issue return code 100.	DMKIMG	LOOP1 LOOP2 PARMERR	
2 Create a DUMMY image library.		ENDPARMS	
3 Issue a FILEDEF command with the AUXPROC option for the DUMMY image library created in Step 2; this traps all READ and WRITE operations on that data set. If any FILEDEF errors occur, issue a return code of 104.		FILEBAD	
4 Issue a GETMAIN for a 73,000 byte area for simulating OS LOAD macros.			
5 Issue HNDSVC macro to handle the following SVCs: • SVC 8 (LOAD) • SVC 18 (BLDL) • SVC 21 (STOW)			
6 Call IEBIMAGE: • Issue a CMS STATE command for the TEXT file being searched for and set appropriate return codes in SVC save area.		BLDLRTN	
• Issue CMS LOAD for requested module and return the address of the area loaded into, to the issuer of the LOAD command.		LOADRTN	
• Use CMS LOAD command to get module into LOAD area and move data into user-supplied buffer for the READ.		READRTN	
• Treat as no-op and return to issuer.		WRITERTN READEXIT	
• Simulate operation of STOW macro by locating the module data in the IEBIBLKS work area and create a TEXT deck from it: 1) Create ESD (external symbol directory) card and write to TEXT file (GENIMAGE CMSUT1)		STOWRTN	
2) Create all necessary TXT cards and write to TEXT file (GENIMAGE CMSUT1)		TXTLOOP	
3) Create an END card and write to TEXT file (GENIMAGE CMSUT1)			
7 Erase old TEXT file (if one existed) and rename GENIMAGE CMSUT1 to a TEXT file named IEBIMAGE.			

Diagram 13-1. DMKIMG



1 Verify existence of control file.

2 Load all TEXT files requested in control file into virtual storage and construct directory entries.

3 Cause loaded file to be stored as the named system.

Notes	Module	Label	Ref
1 Verify the existence of the control file. If it doesn't exist, give a return code of 4.	DMKNMT	IMAGELIB ERR004	
2 Create a DUMMY directory that will be used to hold the number of entries in the named system.			
Read a record from the control file and verify the existence of the indicated TEXT file. If it doesn't exist, issue message DMKNMT248E.		RDLOOP AFTERRD NOTEXT	
Load the TEXT file into the CMS transient area. If a LOAD error occurs, issue message DMKNMT256E.		LDERR	
Move the file from the transient area to the core image area if sufficient storage exists. If not, issue message DMKNMT256E.			
Create a new directory entry for this TEXT file and return to RDLOOP. If no more entries, close the control file, compress the core image, and adjust the displacements in the directory.		RANOUT RDEOF DSPLOOP	

Notes	Module	Label	Ref
3 Issue DIAGNOSE X'74' to cause the named system to be saved. If successful, issue message DMKNMT247I; if not successful, issue message DMKNMT254E.		DIAGERR	

Diagram 13-2. DMKNMT

Program Organization

This section includes program descriptions of modules DMKIMG and DMKMNT.

DMKIMG

Provides a CMS interface for the VS-based IEBIMAGE program by handling certain SVCs issued by IEBIMAGE and translating them into CMS terms.

Entry Point
DMKIMGBG

Routines Called
FSSTATE - Determines if control file exists.
HNDSVC - Traps certain SVCs issued by IEBIMAGE.
GETMAIN - Gets area for simulating OS LOAD SVC.
FREEMAIN - Releases OS LOAD area.
FILEDEF - Issues FILEDEFs needed by IEBIMAGE.
LOAD - Simulates OS LOAD and QSAM READ.
FSWRITE - Creates a new TEXT file (STPW simulation).

Attributes
Disk resident, loaded into CMS user area, called via SVC 202, serially reusuable.

Registers at Entry
R1: Standard CMS PLIST R14: Return address R15: Address of GENIMAGE

Registers at Exit
R15: Return code < 100 for normal IEBIMAGE execution
R15: Return code 100 if error in input parameters
R15: Return code 104 if error during FILEDEF

External References
MAINHIGH - Saves and restores its value between loads.

DMKNMT

Constructs an image library from TEXT files on user disks and creates or replaces that image library via DIAGNOSE code X'74'.

Entry Point
DMKNMTBL

Routines Called
FSSTATE - Determines if CNTRL and TEXT files exist.
FSREAD - Reads in the control file.
CMS LOAD - Loads the TEXT file into the transient area.

Attributes
Disk resident as "IMAGELIB", loaded into CMS user area, called via SVC 202, serially reusuable.

Registers at Entry
R1: Standard CMS PLIST

Register at Exit
Register 15 contains a return code:

Return Code	Meaning
0	Image library updated successfully
4	Control file not found or in error
8	Specified image non-existent
12	Specified image caused LOAD error
16	Insufficient virtual storage
20	Image library is currently active
100+ return code	Error in FSREAD

Register Usage

R0:	Temporary work register
R1:	PLIST register and temporary work register
R2:	Source address for MVCL
R3:	Source length for MVCL
R4:	Target address for MVCL
R5:	Target length for MVCL
R6:	Current end of image library in storage
R7:	Pointer to next available directory entry
R8:	Running counter for number of directory entries
R9:	Starting address of the image library in storage
R12:	DMKNMT module base
R14:	BALR return address and scratch register
R15:	BALR branch address and scratch register

External References
None

Directory

Figure 13-2 is an alphabetical list of the major labels in modules DMKIMG and DMKNMT. The figure indicates the associated method of operation diagrams and it provides a brief description of the operation performed at the point in the program associated with each label.

Label	Diagram	Description
AFTERRD	13-2	Saves the name of the control file.
BLDL2	13-1	Checks for file.
BLDL3	13-1	
BLDRET	13-1	Return to user key.
DIAGERR	13-2	Issue error message DMKNMT254E.
DSPLOOP	13-2	Adjusts old displacement in directory entries.
-ENDPARMS	13-1	Creates DUMMY image library.
ERR004	13-2	Issues return code of 4.
-FILEBAD	13-1	Issues FILEDEF error.
GETSEQ	13-1	Obtains current value of sequence number.
LDERR	13-2	Issues error message DMKNMT249E.
LOADRTN	13-1	Simulates LOAD functions.
LOOP1	13-1	Validates parameter list.
LOOP2	13-1	Validates options.
MOVETXT	13-1	
NOTEXT	13-2	Issues error message DMKNMT248E.
OPTIONS	13-1	Scans through options.
PARMERR	13-1	Gives return code 100 for parameter error.
RANOUT	13-2	Issue error message DMKNMT256E.
RDEOF	13-2	Saves file name for CLOSE.
RDERR	13-2	Checks for end-of-file.
RDLOOP	13-2	Points to file name.
READEXIT	13-1	Issue return codes from READ.
READRTN	13-1	Simulates READ functions.
RETURN	13-1	Saves return code.
RETURN	13-2	Obtains return address.
STOWRTN	13-1	Simulates STOW functions.
TXTLOOP	13-1	Creates TXT cards.
WRITERTN	13-1	Simulates WRITE functions.

Figure 13-2. DMKIMG and DMKNMT Label Directory

Data Areas

The following data areas are used by DMKIMG and DMKNMT:

- Data Control Block (DCB)
- Data Extent Block (DEB)
- Data Extent Control Block (DECB)
- (PDEBLOK)

All the above data areas except PDEBLOK are described in the *OS/VS2 Debugging Handbook, Vol. 2*, Order No. GC28-0988. The PDEBLOK is described in Figure 13-3.

```
 0 ┌──────────────────────────────────────────────────────────┐
   │                         PDENAME                          │
 4 │                                                          │
   │──────────────────────────────────────────────────────────│
 8 │                         PDEDISP                          │
   │──────────────────────────────────────────────────────────│
 C │                         PDELGTH                          │
   └──────────────────────────────────────────────────────────┘
```

Displacement					
Hex	Dec	Field Name			Description
0	0	PDENAME	DS	CL8	Member name
8	8	PDEDISP	DS	1F	RBA of start of member
0C	12	PDELGTH	DS	1F	Length of member in bytes

Figure 13-3. PDEBLOK Directory Entry for Named System

Diagnostic Aids

Figure 13-4 lists the messages issued by and the associated method cf operation the DMKIMG and DMKNMT. The nearest label diagram are identified.

Message Code	Label	Diagram	Message Test
DMKNMT247I	RETURN	13-2	3800 NAMED SYSTEM CREATED SPECIFIED IMAGE image
DMKNMT248E	NOTEXT	13-2	SPECIFIED IMAGE image NON-EXISTENT
DMKNMT249E	LDERR	13-2	ERROR LOADING IMAGE image
DMKNMT254E	DIAGERR	13-2	ERROR SAVING imag3800 - RC = (return code)
DMKNMT256E	RANOUT	13-2	INSUFFICIENT VIRTUAL STORAGE

Figure 13-4. DMKIMG and DMKNMT Messages

Index

READER'S COMMENT FORM

Title: IBM Virtual Machine
Facility/370:
Service Routines
Program Logic

Order No. SY20-0882-4

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:

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

SY20-0882-4

Reader's Comment Form

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

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

Cut or Fold Along Line

SY20-0882-4

IBM Virtual Machine Facility/370: Service Routines Program Logic

Printed in U.S.A.

SY20-0882-4

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