

Computer Management Systems (CMS)

SYSTEM SOFTWARE OPERATION GUIDE

THIS MANUAL REPLACES FORM 2007258 DATED FEBRUARY 1977

(INCLUDES PCN-001)

COPYRIGHT © 1976, 1977 BURROUGHS MACHINES LIMITED, Hounslow, England
COPYRIGHT © 1976, 1977 BURROUGHS CORPORATION, Detroit, Michigan 48232

PRICED ITEM

LIST OF EFFECTIVE PAGES

NOTE: Insert latest changed page;
dispose of superseded pages.

TOTAL NUMBER OF PAGES IN THIS MANUAL IS
76 CONSISTING OF THE FOLLOWING:

Page No.	Issue
Title	September 1977
1-1 thru 1-2	September 1977
2-1 thru 2-5	September 1977
2-6 Blank	September 1977
3-1 thru 3-84	September 1977
4-1 thru 4-17	September 1977
4-18 Blank	September 1977
5-1 thru 5-34	September 1977
6-1 thru 6-41	September 1977
A-1 thru A-10	September 1977
B-1 thru B-16	September 1977

Burroughs believes that the information described in this manual is accurate and reliable, and much care has been taken in its preparation. However, no responsibility, financial or otherwise, is accepted for any consequences arising out of the use of this material. The information contained herein is subject to change. Revisions may be issued to advise of such changes and/or additions.

Burroughs 

Computer Management Systems (CMS)

**SYSTEM SOFTWARE
OPERATION GUIDE**

THIS MANUAL REPLACES FORM 2007258 DATED FEBRUARY 1977

COPYRIGHT © 1976, 1977 BURROUGHS MACHINES LIMITED, Hounslow, England
COPYRIGHT © 1976, 1977 BURROUGHS CORPORATION, Detroit, Michigan 48232

PRICED ITEM

Burroughs



PUBLICATION
CHANGE
NOTICE

PCN No.: 2007258-001 Date: January 1978
Publication Title: Computer Management System (CMS) System Software
Operation Guide
Other Affected Publications: None
Supersedes: _____

Description

This Publication Change Notice incorporates the Table of Contents into the basic manual. Insert pages iii through vi.

Retain this PCN as a record of changes made to the basic publication.

BMG activities may obtain additional copies of this PCN by ordering the above form number from Literature Distribution, Dearborn, Michigan.

International activities requiring additional copies of this PCN should determine their annual requirements and submit them to their literature Coordinator.

LIST OF EFFECTIVE PAGES

Page No.	Issue
Title	September 1977
iii thru vi	January 1978
1-1 thru 1-2	September 1977
2-1 thru 2-5	September 1977
2-6	Blank
3-1 thru 3-84	September 1977
4-1 thru 4-17	September 1977
4-18	Blank
5-1 thru 5-34	September 1977
6-1 thru 6-41	December 1977
6-42	Blank
A-1 thru A-10	September 1977
B-1 thru B-16	September 1977

TABLE OF CONTENTS (Continued)

Section		Page	Section		Page
6	Question-Answer Routine	6-5		Declare (DECL) Registers	6-31
	Prompting Mode Operating Instructions	6-6		Display	6-32
	Prompting Error Messages	6-6		Miscellaneous (MISC)	6-32
	No-Prompting Mode Operating Instructions	6-8		Task Control Block (TCB)	6-32
	No-Prompting Error Messages	6-8		Message (MSG) Reference Area	6-33
	Copy	6-8		Data Segment Table	6-33
	Bootstrap Warmstart	6-10		Control Stack	6-33
	Release Letter	6-10		Data Segments	6-34
	Patch	6-11		File Analysis	6-35
	Warm Start	6-16		Control Stack	6-35
	Warm-Start Operating Procedure	6-16		Registers	6-36
	Restart Procedure	6-18		Descriptor Information	6-36
	Restart Operating Instructions	6-18		COBOL Dump Analyzer	
	Clear Start Procedure	6-18		(COBOLDUMP)	6-36
	System Dump Analyzer (SYSDUMP)	6-21		Heading	6-37
	Heading	6-23		Program Parameter Block (PPB) of	
	patch History	6-23		Code File	6-37
	Hardware Registers	6-24		Task Control Block (TCB) Preset Area	6-38
	Operating System Registers	6-24		Data Segment Table	6-39
	Virtual Memory Links	6-25		Control Stack	6-39
	Peripheral Assignments and Descriptors	6-25		Data Segments	6-40
	Task Detail Table	6-26		Current Operation (COP) Table	6-41
	Print TCB	6-27			
	Print Slices	6-28	APPENDIX A		
	MPLII Dump Analyzer (MPL2DUMP)	6-29		System Control Language Commands	A-1
	Explanation of Formatted Dumps	6-30	APPENDIX B		
	Heading	6-30		System Output Messages	B-1
	S-Registers	6-31			

TABLE OF CONTENTS

Section	Page	Section	Page
1		GENERAL INFORMATION	
		Introduction	1-1
		Master Control Program (MCP)	1-1
		Interpreters	1-1
		Compilers	1-2
		Utilities	1-2
		To the User	1-2
2		DEFINITIONS AND EXPLANATIONS	
		Files	2-1
		Disc and Disc Pack	2-1
		Disc Initialization	2-1
		Magnetic Tape File Names	2-2
		Line Printer and Console File Names	2-2
		Disk File Names	2-2
		Group Names	2-2
		Disk Directory	2-3
		Available Table	2-3
		Program Names	2-4
		Mix Numbers	2-4
		Peripherals	2-4
		Message Syntax	2-5
3		SYSTEM CONTROL LANGUAGE	
		(Refer to Appendix A)	
4		SORT/MERGE INTRINSICS	
		Introduction	4-1
		Related Documents	4-1
		General Description	4-1
		Functional Description	4-2
		Keys	4-2
		Deleted Records	4-3
		Regular Sort Intrinsic	4-3
		Capabilities	4-3
		Complete File Ordering	4-3
		Partial File Ordering	4-3
		Tagfile Creation	4-3
		Keyfile Creation	4-4
		Invalid Index Keys	4-4
		Input Medium	4-5
		Input Restrictions	4-4
		Main Memory Requirements	4-5
		Output Medium	4-5
		Inplace Sort Intrinsic	4-6
		Capabilities	4-6
		Input Medium	4-6
		Input Restrictions	4-6
		Main Memory Requirements	4-6
		Output Medium	4-6
		Merge Intrinsic	4-6
		Capabilities	4-6
		Disk Space Requirements	4-7
		Input Medium	4-7
		Input Restrictions	4-7
		Main Memory Requirements	4-7
		Output Medium	4-7
		Invocation of the Sort-Merge Intrinsic	4-8
		General	4-8
		COBOL Generated Sorts and Merges	4-8
		COBOL Sort Verb	4-8
		COBOL Merge Verb	4-8
		Stand-Alone Sort-Merge Initiation	4-9
		Input Medium	4-9
		Input Restrictions	4-9
		Invoking the Sort Language Processor	4-9
		The Sort Language	4-10
		General	4-10
		The File Statement	4-10
		The Key Statement	4-12
		The User Option Statement	4-13
		Sort Language Processor Messages	4-14
		General	4-14
		Warning Messages	4-14
		Error Messages	4-15
		Sort-Merge Intrinsic Messages	4-16
		General	4-16
		Intrinsic Messages	4-16
		Sort Language Reserved Words	4-17
5		B 80 DEPENDENT ROUTINES	
		Functional Summary	5-1
		CMS B 80 Bootstrap	5-2
		Bootstrap Load	5-2
		Possible Errors	5-2
		Table of Keyboard Indicators	5-3
		Stand-Alone Utility	5-4
		Loading the Utility	5-4
		Operation	5-6
		Functional Descriptions	5-6
		Copy (Stand-Alone Disc Copy)	5-7
		FE (Initialise MTR Disc)	5-9
		IN (Initialise Disc)	5-11
		LD (Load Disc)	5-13
		LS (List Size)	5-15
		OL (Print Status of Cassette Drives)	5-16
		RF (Reference Disc)	5-17
		RL (Relabel Disc)	5-19
		RM (Remove Files)	5-21
		WS (Warm Start)	5-23
		Warm Start Procedure	5-24
		Initiating Warm Start	5-24
		Memory Dump	5-26
		Memory Dump to Disc	5-29
		Customer Confidence Routine	5-31
		Functional Summary	5-31
		Requirements	5-32
		Confidence Routine Operation	5-32
6		B 800 DEPENDENT ROUTINES	
		General	6-1
		Create	6-1
		Set-Up Procedure	6-1
		Operating Instructions	6-1
		Bootstrap Error Handling	6-1
		Error Messages and Recovery	6-2
		Disk Generator	6-3
		General Description	6-3
		Operating Instructions	6-4
		Initialization Routine (INIT)	6-4
		System Disk Generation Routine (DSKGEN)	6-5

SECTION 1

GENERAL INFORMATION

INTRODUCTION

The Computer Management System (CMS) is the realization of a concept on a range of Burroughs hardware equipment. The concept provides for complete portability of source and object programs, and user files, and provides an identical interface between the user and the system, across the range of equipment which supports CMS. The realization consists of a Master Control Program (MCP) which has a well defined user interface, together with System Language Interpreters, and a suite of compilers and utilities. The MCP and interpreters are written in the machine language (micro-code) of the appropriate equipment, whereas the compilers and utilities are written in the Burroughs CMS implementation language (BIL) and interpreted using the appropriate micro-coded interpreter.

The following paragraphs describe briefly the functions of the various components of CMS.

Master Control Program (MCP)

The Master Control Program is the intelligent interface between the user, user programs, system functions, and system resources. It assigns resources to tasks and interleaves task executions allowing multiple tasks to proceed asynchronously as their respective resources become available. The following list shows some of the characteristics of the MCP.

It provides a simple command/response interface between the operator of the system and the system resources.

It provides complete resource management including all Input-Output devices, memory (using Burroughs proven Virtual Memory Techniques), and the processor .

It provides an interface between user programs and Input-Output devices which includes

- Automatic Label Recognition and Maintenance.
- File Handling as defined for COBOL.
- File access methods such as RANDOM, INDEXED SEQUENTIAL.
- Error Handling.

It provides automatic switching from one task to another when the executing task has to wait for an autonomous system function to complete.

Full details of the Master Control Program are to be found in the CMS Master Control Program (MCP) Reference Manual, form number 2007555.

Interpreters

An interpreter is an organized set of micro-coded routines which interfaces a user program with the MCP, and, together with the MCP, performs the hardware actions required to execute the user program instructions (S-codes).

A full description of each CMS interpreter is to be found in the Virtual Machine sections of the CMS Master Control Program (MCP) Reference Manual, form number 2007555.

Compilers

The function of a compiler is to translate a machine-readable copy of a source program into an executable object program, after checking that the syntax of the source is correct. Full descriptions of each CMS compiler are to be found in the following manuals.

CMS COBOL Reference Manual, form number 2007266
CMS RPG Reference Manual, form number 2007274
CMS MPLII Reference Manual, form number 2007563

Utilities

Utilities are provided which allow efficient system, media, and file maintenance. The utilities available with CMS are fully described in this manual.

TO THE USER

This manual is written both as a book to be read from beginning to end to provide an introduction to CMS, and as a reference manual to be consulted from time to time on various points.

As a book, the manual explains how the MCP may be installed on a new machine and how it may be restarted after the machine is switched off. Each system function provided with the machine is described, both how it is used, and why. These sections cover the basic operation of the system and, together with the language manuals, provide all the information required to write, compile and execute programs on the system.

All messages which may be printed by the MCP or by the system functions are listed in alphabetical order together with a description of the message, the reasons for which it has been printed, and, in the case of error messages, the corrective action which can be taken.

If the manual is to be used as a reference manual, it is recommended that the user familiarize himself with section 2, which defines the notation and abbreviations used throughout the manual. For ease of reference, the description of each system function begins on a new page with the function mnemonic printed in bold type at the top of the page. The list of system messages provides a rapid cross reference between system messages and required operator actions. Finally a list of system functions and their various formats is provided in a concise form for ease of reference.

SECTION 2

DEFINITIONS AND EXPLANATIONS

FILES

All information, programs and data, used by the system is manipulated as files. A file is simply a collection of related records. Each record of a file is of the same size. For more efficient use of media, records may be grouped into blocks whose size is always a multiple of the record size. All blocks in a file have the same size. A file is referenced by its file name which must be known to the system before a file can be used. The identifier `< file—name >` is used in this manual to indicate that any format of file—name for a particular device type is permitted in the syntax. The various formats are described below for each device type.

The system regards all input-output operations, including keyboard input and printing, as file input or output. In the case of a printer file, the block size is the same as the record size which is one line. Keyboard or console files have special constructs to enable the reading and writing of particular fields within a record. Magnetic Tape files normally have a special label record at their start to enable the system to determine the file name, record size and block size of the file. It is the systems responsibility to write such a label record at the beginning of a tape being created by a program unless the program requests otherwise. Many files, including all program and system files, are maintained on disk.

DISC AND DISC PACK

A disc consists of one platter, both surfaces of which may be used to record data. A disc pack consists of more than one disc mounted upon the same spindle, which therefore rotate as a unit. The recording area of discs and disc packs is divided into the following physical levels of addressing space.

- Cylinder —An area on both surfaces of the disc (or all surfaces of all discs in a pack) which is at the same radial distance from the centre of the disc.
- Track —An area on one surface of a disc which is at the same radial distance from the centre of the disc. There are $2n$ tracks per cylinder, where n is the number of discs per pack.
- Sector
(or Segment)—The basic unit of disc address. The sector is that portion of a track which is read or written as a unit. One sector contains 180 bytes of data, and several additional bytes used by the disc controller. These additional bytes are not accessible by user programs.

DISC INITIALIZATION

Before a disc can be used with a CMS machine, certain information must be present on the disc. This information cannot be installed during the manufacture of the disc, since the disc may be used on many different systems, with entirely different implementations. The CMS MCP gains access to particular disc sectors through a sector address field in the additional bytes mentioned above, and locates disc files through a disc directory which is maintained on the disc. It is the function of the initialisation stand-alone utility to construct, on a disc of unknown format, the correct sector addresses and directory structure required by CMS. A brief description of the CMS disc directory structure is to be found below, and full details are available in the CMS Master Control Program (MCP) Reference Manual, form number 2007555.

MAGNETIC TAPE FILE NAMES

Note: the term "magnetic tape" is used to include tape cassette unless otherwise stated.

A magnetic tape may be used to store one file (a "single file tape") or more than one file (a "multifile tape"). A single file tape will have a beginning label and an ending label to delimit the file. A file which is too large to be contained on a single reel may extend to subsequent reels, each of which will contain a beginning and an ending label. A multifile tape will have a beginning and an ending label to delimit each file. The last file on a reel may extend to subsequent reels.

Magnetic tape labels contain fields for the multifile-identifier (mfid) and file-identifier (fid) of the file. Each field is seven characters long. The < mfid > in each label of a single file tape will contain "OOOOOOO". The < mfid > of each label of each file of a multifile tape will contain the tape name, (multifile-identifier).

The < fid > of each label of single or multifile tapes will contain the appropriate file-identifier. Refer to the CMS Master Control Program (MCP) Reference Manual, form number 2007555 for full details of Magnetic Tape Labels.

LINE PRINTER AND CONSOLE FILE NAMES

Line printer (which includes serial printer used as line printer) and console files may be labelled. The file-identifier (fid) consists of up to seven characters, and is printed as part of the beginning and ending labels of the file. Refer to the CMS Master Control Program (MCP) Reference Manual, form number 2007555 for full details of these labels.

DISK FILE NAMES

A disk file name has the following format:

< disk-id > / < file-id >

< disk-id > is the name of the cartridge which contains the file. The < disk-id > consists of between one and seven alpha-numeric characters. In some cases, it is not necessary to specify the disk-id and in those cases, the current system disk is assumed to be the one required. Specifying a < disk-id > of 0000000 will imply the current system disk.

< file-id > consists of between one and twelve alpha-numeric characters or hypens. It is used to identify the particular file on the specified disk. All files on a particular disk will have unique file-names; this is enforced by the MCP. A full description of the Disc Cartridge Label (which contains the disk-id) and the Disc Directory (which contains the file-ids) is to be found in the CMS Master Control Program (MCP) Reference Manual, form number 2007555.

GROUP NAMES

In some utilities it is possible to perform a function on a group of disk files. Such a group specification has the following format:

< disk-id > / < group-id > =

< group-id > consists of between zero and eleven alpha-numeric characters or hypens and specifies the characters which define the group. All files on the specified disk (groups do not extend over more than one disk) whose < file-id's > begin with the characters specified in < group-id > are considered members of the group.

Examples:

If a disk contains files with ids

A,ABC,ABCD,BCD,BCDE

then the following < group—ids > refer to these files

=	A =	AB =	ABC =	BC =
A	A	ABC	ABC	BCD
ABC	ABC	ABCD	ABCD	BCDE
ABCD	ABCD			
BCD				
BCDE				

Specifying a < group—id > with no characters refers to all files on the specified disk. Also, different < group—id >'s may give rise to the same group (see AB = and ABC = above).

The use of the file-group is most effective if the creation of < file—id >'s is given careful thought: for example, in a payroll suite, all files may be given < file—ids >'s beginning "PAY". Program files within the suite could begin "PAYP", with "PAYPS", and "PAYPC" designating source and code files. Data files for use with the suite could begin "PAYD", with further identification for different types of data files, for example "PAYDT" for transaction data. Thus the whole payroll suite, all data, source and object programs be referenced by the group "PAY =", all programs by the group PAYP = and all source programs by PAYPS = . The SCL command RM PAYDT = will remove all transaction files of the payroll suite.

DISK DIRECTORY

The disk-directory is a table which contains an entry which describes each file on the disk. In particular, the name of each file and where on the disk its information is stored, is kept in the directory. Thus, whenever a program requires to read a file, the directory on the specified disk is searched for the required < file—id > and the directory entry points to the information required. Whenever a file is to be created, an available entry is found in the directory and this is filled with information related to the new file.

The directory on a CMS disk is a fixed size determined at disk initialization time and any attempt to create more files than there are entries in the directory will cause an error message to be printed.

The disk space required for the directory is allocated by the initialize stand alone utility, which also makes all entries in the directory available.

AVAILABLE TABLE

The available table enables the operating system to find available space on the disk whenever new files are created or files are extended. If there is insufficient available space on the disk, a message is printed and the operator may remove files from the appropriate disk in order to make space available. The KA utility provides information as to available space on the disk. When a disk is initialized, the available table built by the initialize stand—alone utility indicates that the whole disk except the directory and other system reserved areas is available space.

PROGRAM NAMES

A "program" is a special disk file whose type is "CODE". The program name is the < file-id > of this program file. This program file is generated by a compiler, which translates the program source code into the machine understandable form of a program file. In order to execute this program, it is necessary to "load" it, that is, build the tables and structures required by the MCP to maintain control of the program and hence the total system. Compilers translate their source programs into an S-code and this S-code must be interpreted on the hardware in order to execute the program. This interpretation is performed by an interpreter which depends upon the language being executed. COBOL and RPG programs are both interpreted by the COBOL S-CODE interpreter called "COBOLINT", MPL programs are interpreted by the BIL S-CODE interpreter called "BILINTERP". Both interpreters will appear as files on a disk containing the complete system, and their type will be "INT80" (which will appear as "SYSTEM" in a listing produced by the LR utility). If it is never required to run MPL programs then "BILINTERP" may be removed and similarly if neither COBOL or RPG programs will be run, the "COBOLINT" file may be removed.

MIX NUMBERS

As a program is loaded, it is assigned a mix number. This is the system's unique identifier for the current execution of the program. Mix numbers will cycle from 1 thru 9 and back as programs are executed. All messages to or from the operator concerning any program will always quote the mix number. Messages to the operator from the system will also always quote the program name which is optional in most messages from the operator but if specified must be correct.

PERIPHERALS

In some cases, it is necessary for the operator to indicate to the system a particular peripheral unit, for example, to request the status of a particular unit. The specification of a peripheral unit is via a three character mnemonic where the first two characters specify the peripheral type and the last character specifies the particular unit on the users system. Thus DK is the mnemonic for a cartridge disk and the first drive will be specified by DKA, the second by DKB etc. The various peripheral types are listed below:

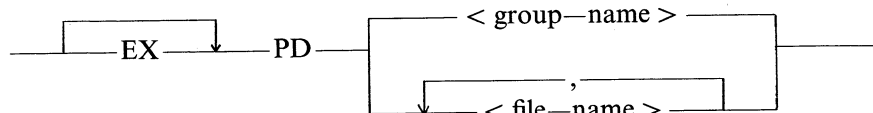
R8	80 column reader
R9	96 column reader
P8	80 column punch
P9	96 column punch
M8	80 column reader/printer/punch
M9	96 column reader/printer/punch
AR	Any card reader
CP	Any card punch
AM	Any reader/printer/punch
AC	Console with any output device
PC	Console with serial printer
SC	Console with self-scan
✓ AP	Any record printer
✓ SP	Serial printer (used as record printer) <i>console printer</i>
✓ VLP	Line printer
AT	Any magnetic tape
✓ MT	Magnetic tape (reel)
✓ CT	Cassette tape
✓ DK	Disk cartridge (any type or speed)
DP	Disk pack
✓ HT	Disk Head per Track
✓ DF	Burroughs Super Mini disk <i>Fixed Disk</i>
✓ DC	Data Comm
✓ SS	<i>SELF SCAN DISPLAY</i>
✓ AN	<i>CONSOLE KEYBOARD</i>
✓ DM	<i>MINI DISK</i>

In some messages, only certain types of peripheral specification are permitted. For example, the PO function is only applicable to disks. For such cases, the construct "< disk peripheral >" or "< cassette peripheral >" will be used to indicate a subset of the full < peripheral > specification.

MESSAGE SYNTAX

The complete syntax for each system function is described at the beginning of each description as a "railroad" diagram. These diagrams are reproduced at the end of the manual as a quick reference guide to the System Control Language (SCL). The railroad diagram provides a compact and easy to use definition of the syntax of each SCL command and it allows the operator to make a rapid check of the syntax of any doubtful message.

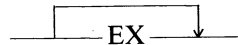
Below is the railroad diagram for the syntax of the PD function described in Section 3 of the manual. Here we are concerned only with the interpretation of the diagram.



The diagram should be read from left to right unless flow arrows indicate differently. By typing in or checking each unit as it is encountered, an SCL command may be constructed or validated.

The diagram should be read from left to right following the lines or railroad. All valid variations of a command may be generated by starting on the left, typing each unit as it is encountered until the end point on the right is reached.

In the PD diagram above, the first construct encountered is:



this indicates that, syntactically, "EX" is optional as the first characters of this SCL command.

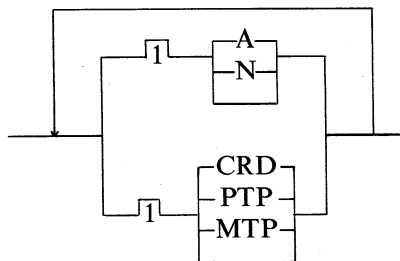
The next construct



indicates that "PD" must occur as the next two characters in the command. The track then splits indicating another option. Taking an overall view of the diagram, it can be seen that the two options available at this point do not rejoin until the end point, indicating two formats of the PD command. On the upper track a single < group-name > is required and on the lower track, the construct

< file-name >

indicates that at least one < file-name > must be specified. By following the return track of this construct, a comma must be the next character and the lower track is rejoined before the file-name specifier. This construct then indicates a list of < file-names > separated by commas and containing at least one file name. A number in a loop of a track indicates the maximum number of passes through that track, for example:



indicates that "CRD A" and "A CRD" are valid constructs whereas "A A" or "A CRD N" are not valid constructs.

When following a railroad diagram, multiple spaces are allowed between each syntax element, and at least one space should be typed between syntactic elements that are not recognised delimiters (for example, comma).

SECTION 3

SYSTEM CONTROL LANGUAGE

Communication with the MCP is through the console keyboard and printer. The MCP prints its messages on the left hand tractor of multiple tractor consoles, otherwise, the leftmost 50 character positions of a solitary tractor are reserved for system communication. This system communication area is called the system journal or "SPO". In order to prepare the MCP to accept command input, the Ready Request key must be pressed. If the MCP is able to accept input it responds by lighting the ready and alpha lights and extinguishing all others. If the system is unable to accept input because, for example, it is loading a program or performing functions previously typed in, then the Ready Request key is ignored and the operator should press it every few seconds until a response is obtained. When the alpha and ready lights are lit, all keyboard input is directed to the MCP as SCL commands. Input on the keyboard is echoed immediately, but if the console printer is being used by an executing program, then the current output to the console is completed before the print head is released to echo SCL input. Termination of an SCL command is by an OCK. This terminator is not recorded as part of the command. While typing commands, the backspace key may be used to erase characters typed in error and the whole message which was typed may be voided by typing RESET. The numeric keyboard is not enabled during SCL input and using it or any invalid alpha key will cause the error light to be lit and the alarm to be sounded. This error may only be cleared by typing RESET which allows continued typing of the command. All messages from the system to the operator are indented three spaces and echoing of operator input begins in the first character position. Note: messages from the system (as distinct from a utility—see below) contain an "event number" enclosed in square brackets. A glossary of event numbers and their meanings is to be found in Appendix B.

The MCP contains certain embedded functions which are invoked through the SCL handler. These functions are concerned with intimate details of the system, for example, the MX and OL commands, and are known as "intrinsic". The SCL handler recognizes these functions, when requested, by their unique mnemonics. Any character string not recognized as an intrinsic is assumed to be a program name, and the loader is invoked. Refer to the description of the EX intrinsic for details of program initiation.

Additional non-intrinsic utilities are available for use with the system. These utilities exist as BIL programs, but may be viewed as extensions to the intrinsic utility functions, since the initiating procedure is identical when the utility is present on the system disc. Examples of these programs are LIST and KA. In the following alphabetically ordered descriptions, the terms "intrinsic" and "utility" are used to distinguish the two types.

The SCL handler is able to pass characters following the program name in an SCL message to the program, as "Initial Parameters" (or "Initiating Message"). This ability is taken advantage of by the BIL utilities, allowing an entire task specification to be input as a single SCL message.

All BIL utilities provide a "macro—call" facility which permits a frequently used task specification to be input from a file on disc. The utilities are directed to the file by using the construct

_____ * _____ < disc—id > / _____ < file—id > _____

at the appropriate point in the initiating message as shown in the following alphabetically ordered descriptions. The file containing the parameters must exist as a maximum of five 80 character records. No nested macro-calls are permitted, that is, a file which is to be used as an * < file—name > must not contain the * < file—name > construct. If the specified file cannot be found the message < file—name > NOT FOUND will be displayed by the utility.

Note: it is worth emphasizing that a BIL utility, since it is an S-code program, may be executed from any disc on the system which contains a copy of the program code—file. This capability may become necessary if the system disc becomes too full to allow execution of any program. The system disc may then be tidied by executing the "remove" utility, RM, from an alternative disc. Refer to the description of the EX intrinsic for details of program initiation.

AD

AD (Assign Peripheral)

_____AD_____ <mix #> / <prog-id> <peripheral> _____

This intrinsic command may be used to assign a particular peripheral to a program that has called for an unlabelled input file, or has requested a line printer containing special forms.

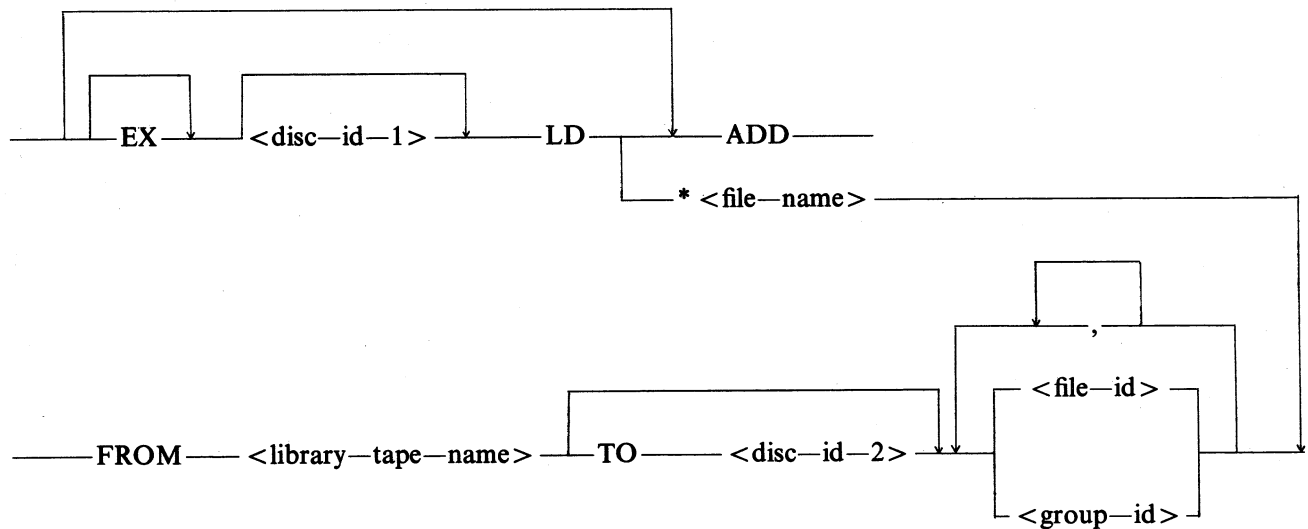
The system informs the operator of the requirement for an unlabelled file by a message of the following format:

<mix-number> / <program-name> [14] WAITING UNLABELLED <device-mnemonic>
DEVICE REQUIRED

The command must only be used when the named program is suspended, waiting for the appropriate type of peripheral. At any other time, the message

<mix-number> / <program-name> AD INVALID

is displayed.

ADD (Add Files from Library Tape)

Note that this function is a sub-program within the utility LD. MCP recognises the mnemonic ADD if LD is not specified, and will automatically initiate a load of the LD utility. The < file-id > of LD should not therefore be changed if the ADD function is to be invoked in this manner. To discontinue the ADD function, "DS < mix-number > /LD" must be used.

This function may be executed from a disc other than the system disc by specifying LD and < disc-id-1 > .

The function provides the capability of copying files or groups of files from a "library tape" to the disc specified by < disc-id-2 > (or the system disc if < disc-id-2 > is not specified). A library tape is defined to be a tape output by the DUMP or UNLOAD functions. The < library-tape-name > identifies the library tape containing the files or groups of files to be copied to the disc. The file list identifies the particular files or groups of files to be copied from the library tape. Note that the file list may contain a mixture of files and groups of files. Only the files which do not have copies already on the disc are loaded.

Output Messages

The following messages will be output by the utility in the event of the corresponding error or condition.

Errors in Initial Parameters

The following messages are output, and the utility will terminate, if the initial parameters are incorrect or invalid.

1) ILLEGAL PARAMETER LIST

This message is displayed if the parameters supplied in the input message are incorrect. The utility will attempt to follow the message with a character string from the area of the input message which caused the failure.

2) INVALID CHARACTER IN IDENTIFIER < identifier >

This message is displayed if the < library-tape-name >, < disc-id >, or < file-id > entries in the input message contain characters which are illegal in identifiers.

3) < file-name > NOT FOUND

This message is displayed if the function was invoked through the LD * < file-name > format, and the parameter-file cannot be located.

4) DISK < disc-id > NOT AVAILABLE

This message is displayed if the specified disc is not on line.

5) NO SPECIFICATION GIVEN

This message is displayed if no parameters are input following LD in the SCL input string.

Normal Execution

The following message is displayed for each file which is added:—

< file—name > LOADED

Errors During Execution

The following messages are displayed if the utility cannot perform some part of its function. The utility will continue to execute wherever possible.

1) < library—tape—name > NOT A RECOGNIZED DUMP TAPE

This message is displayed if the tape which has been provided for loading does not have a recognizable header. The correct tape should be provided, or the utility should be DS'ed (discontinued). If the tape is provided, the GO command may be required in some CMS implementations in order to restart the utility.

2) NO FILES IN THE FAMILY < group—id > ON TAPE < library—tape—name > FOR ADD

This message is displayed if the utility is unable to find any files in the specified group on the library tape.

The utility will continue with the next file in the input parameters list.

3) NO FILE < file—name > ON TAPE < library—tape—name > FOR ADD

This message is displayed if a specified file is not present on the library tape. The utility will continue with the next file in the input parameter list.

4) < file—name > LOAD/DUMP DISCREPANCY

This message will be displayed if end of file has been reached before it is expected, and implies erroneous information in the Disk File Header. The utility will continue with the next file in the input parameter list.

5) NO FILES TO LOAD

This message is displayed if no files will be loaded from the tape. The utility will terminate.

6) < file—name > LOADED

This message will be displayed for each file which is added from the tape.

7) < file—name > NOT LOADED—ALREADY ON DISK

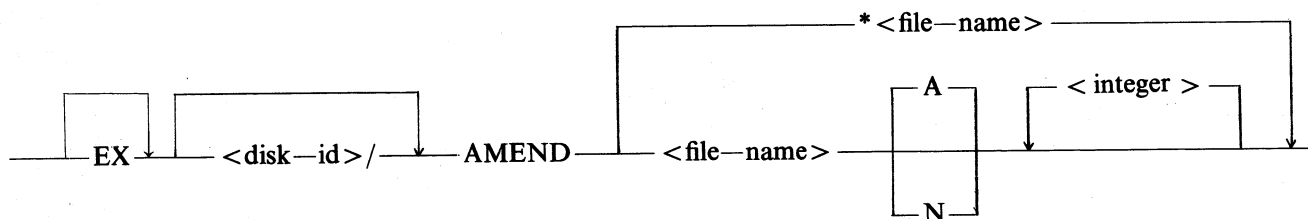
This message will be displayed for each tape file not loaded because a file with the same identity already exists on disc.

8) ALTHOUGH WITH DIFFERENT ATTRIBUTES

This message will immediately follow message 7 above if the two files differ in record size, block size, file size, or file type. This is for information only.

Note that the above messages are output by the utility. Any applicable system message may be output by the MCP (for example ERROR WHILE IN < verb > if the tape cassette drive is opened while in use).

AMEND (Disk file amending)



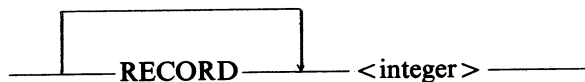
This utility may be used to modify records within an existing file using data input through the console keyboard. The utility may be executed from a disc other than the system disc by specifying < disc-id >. The utility can only be used with systems incorporating a console keyboard. No new file is created. The file to be modified is < file-name >, and it must be a file of type "source" or type "data". If the file is a data file, then input may be specified to be alphanumeric (A) or hexadecimal (N). The default input type is alphanumeric (A). Source file input and Data file type A input is accepted as direct keyboard input, whereas type N will require the input of two characters (0-9, A-F) for each byte of the record.

The integer list specified in the syntax may be used to provide "tab" positions within the record. The use of OCK1 when keying input data causes the utility to reposition the input point in the record to the next tab position. During this repositioning, the utility will fill all character positions left unspecified in the record with a fill character determined by the input type. For source input, the fill character will be an ASCII space, for alphanumeric input an ASCII zero, and for hexadecimal input a binary-zero-filled character. The record length plus one is used as a terminating tab position (whether or not other tabs are specified). The utility can be used for record sizes up to 500 bytes, but, since the utility cannot be given input greater than the width of the console, tab positions are mandatory on files of larger record sizes. For example, a file of 180 byte records requiring alphanumeric input will require at least one tab position (for instance at position 100), whereas a file of 180 byte records requiring hexadecimal input will require a minimum of two tab positions (for instance at positions 60 and 120).

The utility operates in two modes—Record Modify Mode (entered from PK2) or Record Select Mode (entered from PK3). The PK associated with the currently active mode is disabled (the light is turned off) to indicate the appropriate mode. PK1 and PK6 are enabled at appropriate points in either mode to a) select and print the next sequential record from the file (PK1), b) close the file and terminate the utility (PK6). When execution of the utility begins, Record Select Mode is automatically entered.

PK3 Record Select Mode

The required record is identified by logical record number using the following syntax



Where < integer > may take any value from 1 to the number of records in the file.

The required record is located and printed on the console. The print format used corresponds to the input mode selected, that is alphanumeric, source, or hexadecimal. Record Modify Mode may then be entered to alter the data in the record.

PK2 Record Modify Mode

The point in the record at which alterations are to be made is selected by presenting an identifying character string which immediately precedes the required byte of the record.

The character string for insertion or replacement follows the identifying string, delimited by colons (:). If alterations

are to be made at the beginning of the record, no identifying string is input. The syntax of the keyboard input is:—

$\boxed{\langle \text{identifying—string} \rangle}$: $\langle \text{modifying—string} \rangle$:

The use of OCK1 or OCK2 to terminate the input determines whether the $\langle \text{modifying—string} \rangle$ will replace or be added to the existing characters in the record.

OCK1—Replacement

The $\langle \text{modifying—string} \rangle$ replaces the corresponding number of characters in the record if OCK1 is used to terminate the input. For example, with a record containing

ABCD0123

the amendment C:XY: would result in a record containing

ABCXY123.

OCK2—Insertion

The $\langle \text{modifying—string} \rangle$ is inserted at the indicated point if OCK2 is used. The insertion can cause characters in the record to be moved to the right. The shifting of characters applies only to those characters from the starting byte to the next higher relevant tab position; characters beyond this tab position will not be affected.

For example, with a record containing

ABCDEFGH1234

the amendment C:WXYZ: would result in the record

ABCWXYZD1234

if the tabs specified were 7 and 9, or the record

ABCWXYZDEFGH

if the tab specified was 7 alone.

On completion of a modification, the utility will print the amended record with its associated record number, and then illuminate all other usable PK options for possible selection.

When all modifications have been completed PK6 will take the utility to end—of—job.

PK1 Next Record Select

PK1 will cause the next record in the file to be selected and printed. The print format is identical to that provided by PK3. The use of PK1 terminates the Record modify and Record Select modes, therefore a reselection of mode must be made before continuing.

PK6 Terminate

PK6 will close the disk and console files, and cause orderly termination of the utility.

Output Messages

The following messages will be output by the utility in the event of the appropriate error.

Errors in Initial Parameters

These errors occur if the initial parameters following "AMEND" in the SCL input string are incorrect. After the message is displayed, the utility terminates.

- 1) **FILE TYPE IS NOT SOURCE OR DATA**
This message is displayed if the file identified for amendment is of an incorrect file-type.
- 2) **< file—name > NOT FOUND**
This message is displayed if the file identified for amendment or the parameter—file identified by the * < file—name > macro—call cannot be located.
- 3) **ILLEGAL PARAMETER LIST—ATTRIBUTE SPECIFICATION INVALID**
There is an incompatibility between the specified record and block sizes.
- 4) **ILLEGAL PARAMETER LIST—TABS ERROR**
If tab positions beyond the end of the record are specified, or imply input fields larger than the capability of the console, this message is displayed.
- 5) **ILLEGAL PARAMETER LIST**
The utility will attempt to follow this message with a number of characters from the area of the input message which caused the failure.
- 6) **INVALID CHARACTER IN IDENTIFIER < identifier >**
This message is displayed if the < disk—id > or < file—id > portions of the file—name contain characters which are illegal in file—names.
- 7) **NO SPECIFICATION GIVEN**
This message is displayed if no parameters are entered following AMEND in the SCL input string.

Errors During Execution.

The following errors may be encountered during execution of the utility and do not cause the utility to terminate. The messages are displayed via the console file.

- 1) **NOT HEXADECIMAL CHARACTER INPUT—RESUBMIT**
If the input mode is hexadecimal, any character other than 0—9 and A—F will cause the above message to be displayed. The complete entry must be resubmitted.
- 2) **ODD NUMBER OF HEXADECIMAL CHARACTERS INPUT**
If the input mode is hexadecimal, and the number of hexadecimal characters is odd, then this warning is displayed. The utility processes the hexadecimal string by extending it on the right with a zero character.
- 3) **INPUT ERROR—RESUBMIT RECORD MODIFICATION**
If the syntax of the keyboard input in Record Modify Mode is incorrect, the above message is displayed. The entire entry is discarded.
- 4) **BYTE WITHIN RECORD SPECIFIED NOT FOUND**
The < identifying—string > in the keyboard input in Record Modify Mode could not be found in the record. The entire entry is discarded.
- 5) **RECORD SELECTION ERROR**
The above message is displayed if, in Record Select Mode, an invalid record request is made.
- 6) **ILLEGAL RECORD NUMBER SPECIFIED**
This message is displayed if the record number specified in Record Select Mode is greater than the number of records in the file, or is zero.
- 7) **UNWANTED KEY PRESSED—PLEASE RE-INPUT**
This message is displayed if an inappropriate OCK is used to terminate a record.
- 8) **INPUT IMMEDIATELY BEFORE PK6 HAS BEEN LOST**
This warning message is displayed if characters were input immediately before PK6 was pressed to terminate the utility. The characters will not be written to the output file, and the utility will terminate normally.

Note that all the above messages are generated by the utility. Any applicable system message may be output by the MCP in addition to the above (for example, NO FILE if the console is in use).

AX

AX (Accept a message for a task)

— AX — < mix # > / — < prog-id > / — < text > —

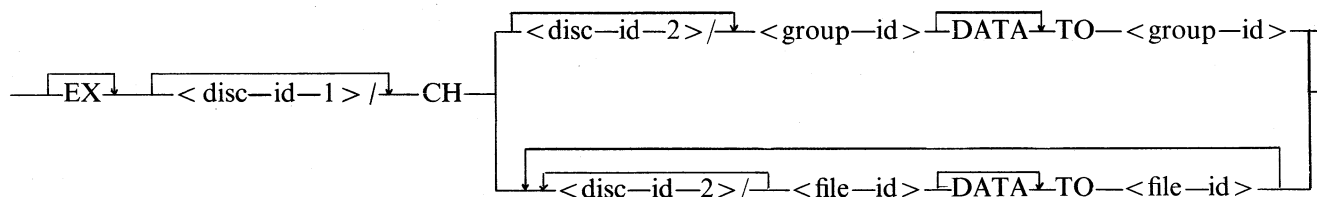
This function, implemented as an intrinsic within the MCP, allows the operator to communicate with a task in the mix. The program must already be suspended waiting for an accept, that is, a message of the format

< mix—number > / < program—name > ACPT

has been displayed by the system. The maximum length of the text is 50 characters and any characters in excess of the length requested by the program are truncated. The input buffer is binary zero filled before accepting characters. If the specified program is not waiting for an accept from the operator or if the mix number and prog-id if specified do not match, then the message

< mix # > / < prog-id > AX INVALID

is displayed.



This utility allows the file identifiers of disc files to be changed, either individually or as one of a group of files. The utility may be executed from a disc other than the system disc by specifying < disc-id-1 >.

If the group change is specified, for each file in the source group, the < group-id > part of the < file-id > is changed to the destination < group-id >. Each individual file-id generated is checked for legality: illegal changes are not performed.

If the file list is specified, each file-id which is found is changed as requested. Each change is checked for legality: illegal changes are not performed.

If the utility detects that a key file has been submitted, and DATA has been specified, then the utility will not change the name of the key file. In this case, the name of the data file to which the key file points will be changed, and the utility will update the pointers in the key file to reference the renamed data file. This change is only performed if both files are on line.

Output Messages

The utility will output the following messages as appropriate during execution.

Errors in Initial Parameters

These errors occur if the initial parameters following "CH" in the SCL input string are incorrect. After the message is displayed, the utility terminates.

- 1) **INVALID CHARACTER IN IDENTIFIER < identifier >**
This message is displayed if the < disk-id > or < file-id > portions of the file-name contain characters which are illegal in file-names.
- 2) **ILLEGAL PARAMETER LIST**
The utility will attempt to follow this message with a number of characters from the area of the input message which caused the failure.
- 3) **< file-name > NOT FOUND**
This message is displayed if the utility cannot find the parameter-file identified by the * < file-name > macro-call.
- 4) **NO SPECIFICATION GIVEN**
This message is displayed if no parameters are entered following CH in the SCL input string.

Normal Execution

For each file-name changed, the utility will output the message

< file-name > CHANGED TO < file-name >

Errors During Execution

The following errors may be encountered during execution of the utility. The file-name related to the errors will not be changed, but the utility will continue to execute wherever possible.

- 1) **NO FILES FOUND FOR CHANGING IN THE FAMILY < group-id >**
The utility will display this message and terminate if the specified < group-id > cannot be located.

- 2) < file—name > NOT CHANGED—NOT FOUND
This message is displayed if a specified file cannot be located. The utility will proceed to the next requested change, or terminate (if no further changes are outstanding). If a file is partly on line, this message will also be displayed.
- 3) < file—name > NOT CHANGED IN USE
This message is displayed if a specified file is in use.
- 4) < identifier > FILE IDENTIFIER TOO LONG
If, when changing a < group—id > to a longer < group—id >, an individual file—name is generated which is longer than the permissible twelve characters, the above message is displayed. The file—name is not changed, and the utility proceeds to the next requested change, or terminates (if no further changes are outstanding).
- 5) < file—name > NOT CHANGED—ILLEGAL REQUEST
If, when changing a < group—id > to another < group—id >, an individual file—name is generated which is "SYSMEM" (a name reserved for system use) or all spaces, the above message is displayed. The file—name is not changed and the utility proceeds to the next requested change, or terminates (if no further changes are outstanding).
- 6) < file—id > NOT CHANGED—< file—id > ALREADY ON DISK
This message is displayed for each change requested which would have generated a duplicate file audition.
- 7) DISK < disc—id > NOT OPENED—NOT ON LINE
This message is displayed if the disc specified in a group change specification is not available. The utility will terminate.
- 8) KEYFILE < file—id > NOW POINTS TO DATA FILE < file—id >
This message is displayed for information purposes when a key file is updated in response to a change specifying DATA.

Note that all the above messages are generated by the utility. Any applicable system message may be output by the MCP in addition to the above (for example, DUPLICATE FILE).

```

┌──EX──┐ ┌──────────────────┐ ┌──────────────────┐ ┌──WITH──┐ ┌──────────────────┐
│      │ │ <disc-id-1> /- │ CHECKADUMP │ <library-tape-name> │ │      │ │ <disk-id-2> │
└──┬──┘ └──┬──┘ └──┬──┘ └──┬──┘ └──┬──┘

```

This utility will compare information in the files on a Library Tape (that is, a tape produced by the LD utility) with the corresponding files on a disc. The utility may be executed from a disc other than the system disc by specifying < disc-id-1 >. The utility may be used to verify that a Library Tape is correct after files have been DUMPed or UNLOADed, or that disc files are correct after files have been ADDED or LOADed. The tape identified by < library-tape-name > is processed sequentially, file by file, and the disc identified by < disk-id-2 > (or the system disc if < disk-id-2 > is omitted) is searched for corresponding files. The utility will comment on up to four errors in a given file. If there are more than four errors, it will ignore the rest of that file, and proceed to the next file on the tape.

Output Messages

The utility will output the following messages as appropriate during execution.

Errors in Initial Parameters

- 1) < parameter-list > ILLEGAL PARAMETER LIST
This message is displayed if the initial parameters following "CHECKADUMP" in the SCL input string are incorrect. The < parameter list > portion of the message attempts to isolate the area containing the error.
- 2) INVALID CHARACTER IN IDENTIFIER < identifier >
This message is displayed if the character strings representing < library-tape-name > or < disc-id-2 > contain characters which are invalid in identifiers.
- 3) NO SPECIFICATION GIVEN
This message is displayed if no parameters are entered following CHECKADUMP in the SCL input string.

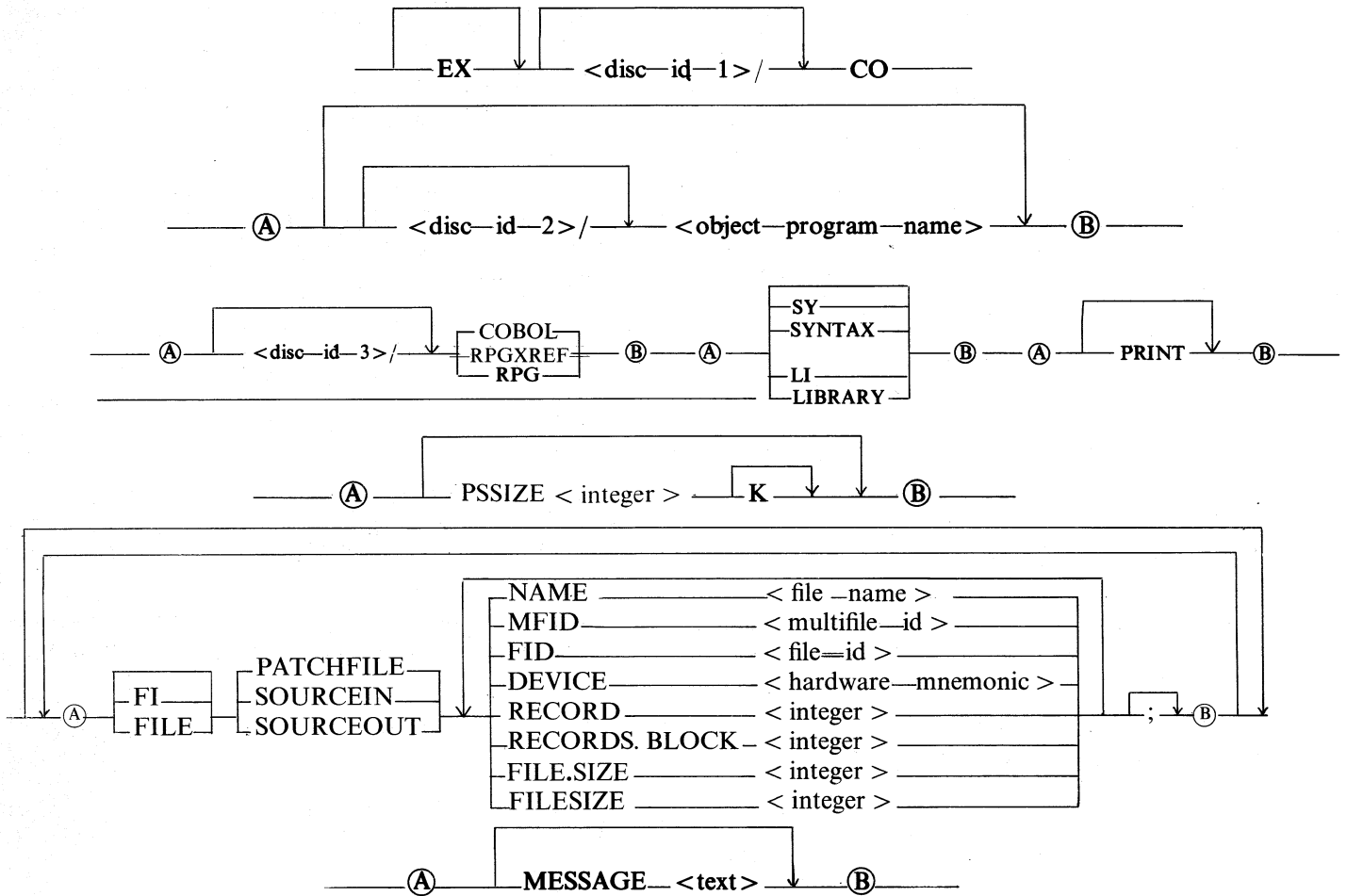
Errors in Execution

- 1) < library-tape-name > NOT A RECOGNISED DUMP TAPE
This message is displayed if the first record of the tape < library-tape-name > is not recognised. The utility will terminate.
- 2) < library-tape-name > INVALID DIRECTORY ON TAPE
This message is displayed if there are more or less entries in the directory on the tape than are specified in the first record of the tape. [Refer to the CMS Master Control Program (MCP) Reference Manual, form number 2007555 for a description of the Library Tape format] The utility will then terminate.
- 3) COMPARISON ERROR ON < library-tape-name > ON DISK FILE HEADERS
This message is displayed if the header in the body of the tape < library-tape-name > is not identical to the respective header in the directory. One is added to the error count for that file. The utility will continue to execute.
- 4) COMPARISON ERROR ON < file-id > FILE NOT FOUND FOR CHECK
This message is displayed if, for a file on the tape, the corresponding disk file cannot be found. The utility will then advance to the next file on the tape.
- 5) COMPARISON ERROR ON < file-id > FILE NOT AVAILABLE FOR CHECK
This message is displayed if, for a file on the tape, the corresponding disk file cannot be successfully opened. The utility will then advance to the next file on the tape.
- 6) COMPARISON ERROR ON < file-id > (AROUND RECORD < number >)
This message is displayed if the utility finds a discrepancy between the tape file and the disc file. The utility will print out a record number in the vicinity of the error in the file, if it is possible for this to be done. One is added to the error count for that file,
- 7) COMPARISON ERROR ON < file-id > AROUND END OF FILE
This message is displayed if there is found to be a difference in the length of the tape and disc files. One is added to the error count for that file.

End-of-Job Messages

- 1) **NO DISCREPANCIES BETWEEN DUMP TAPE < library—tape—name > AND DISK < disk—id >**
This message is displayed if no discrepancies were found between the files on tape and the files on disk.
- 2) **DISCREPANCIES FOUND BETWEEN DUMP TAPE < library—tape—name > AND DISK < disk—id >**
This message is displayed if any discrepancy was discovered between the respective tape and disc files.

Note that all the above messages are generated by the utility. Any applicable system message may be output by the MCP in addition to the above (for example, NO FILE if the required Library Tape is not present).



This utility provides a standard procedure for initiating COBOL and RPG compilations. The utility provides common syntax to be used to define and modify the operating environment of the compiler. The utility may be executed from a disc other than the system disc by specifying `<disc-id-1>`.

The order in which the syntactic elements are described reflects the order in which they are acted upon during a compilation, after the syntax of the message is determined to be correct.

The PRINT clause will cause an analysed listing of the input message to be printed on a line printer (or the console printer if no line printer is available). The analysis includes the information passed to the compiler through a "parameter file" to provide the file equate function. Note: the PRINT clause will be acted upon, if detected, when the syntax of the message is incorrect. Syntax errors prior to the position of the PRINT clause in the message will normally prevent detection of the PRINT clause.

COBOL, RPG or RPGXREF must be specified as shown in the syntax. COBOL and RPG initiate the appropriate compiler, whereas RPGXREF will initiate the cross reference program (RPGXREF) for RPG programs. The compiler may be executed from a disc other than the system disc by specifying `<disc-id-3>`. Note that all passes of the specified compiler, and its global files, must be present on the same disc. The utility will build the appropriate parameter file, ZIP the first pass of the specified compiler or RPGXREF (if no syntax errors are detected in the message), and terminate.

The FILE clauses provide the file equate function. Two input files (PATCHFILE and SOURCEIN) and one output file (SOURCEOUT) may be equated to any file-name, or any device-type available on CMS systems

through the attribute clauses (MFID, FID, DEVICE, RECORD, RECORDS, BLOCK, and FILE.SIZE). These clauses are described below. Note that RECORD, RECORDS.BLOCK and FILE.SIZE are only applicable to the output file identified by the SOURCEOUT clause. The file identified by each FILE clause will be referred to as [PATCHFILE], [SOURCEIN], and [SOURCEOUT].

[PATCHFILE] is the primary input file for the compilation. If all source statements of the source program are contained in one file, then the compiler is directed to this file through the PATCHFILE clause.

[PATCHFILE] may be merged with [SOURCEIN] under control of option records in [PATCHFILE]. When [PATCHFILE] and [SOURCEIN] are merged, records in [PATCHFILE] will override corresponding records of [SOURCEIN]. [SOURCEOUT] may be produced under control of option records in [PATCHFILE]. [SOURCEOUT] will contain the merged contents of [PATCHFILE] and [SOURCEIN] when two input files are used. Refer to the appropriate language reference manual for details of the option records available when using each compiler.

The NAME clause is valid for all files. The < file—name > entry permits MFID and FID to be entered in standard CMS format as follows:

— $\overline{\text{< mfid > /}} \text{< fid >}$ —

If < mfid > is omitted (where applicable) the default "0000000" is used.

The MFID clause is valid for disc and magnetic tape files. The clause allows the < disc—id > or < tape—name > of a file to be defined. The default is "0000000", that is the system disc for disc files, or a single—file tape for magnetic tape and cassette. Note that the use of NAME (above) allows MFID and FID to be entered as a single entry.

The FID clause is valid for all files. The clause allows the < file—id > of a file to be defined. The default is defined in the appropriate language reference manual. Note that NAME (above) allows MFID and FID to be entered as a single entry.

The DEVICE clause is valid for all files. The clause allows the device—type on which the file resides to be defined. The allowable < hardware—mnemonics > are any one of the following:

CR	80 or 96 Column Card Reader
CP	80 or 96 Column Card Punch
CR80	80 Column Card Reader
CP80	80 Column Card Punch
CR96	96 Column Card Reader
CP96	96 Column Card Punch
CRP80	80 Column Card Multifunction Unit
CRP96	96 Column Card Multifunction Unit
CS	
CASSETTE	Magnetic Tape Cassette
MT	Magnetic Tape (Reel to Reel)
DC	
DISK	Any Disc

The default is DISK.

The RECORD clause is valid only in the FILE SOURCEOUT clause. The clause permits the record size of the output file to be defined. The default is 80 characters. A warning is issued if the default is used.

The RECORDS.BLOCK clause is valid only in the FILE SOURCEOUT clause. The clause allows the blocking factor of the output file to be defined. The default is nine(9). A warning is issued if the default is used.

FILE.SIZE and FILE SIZE are synonymous.

The clause is valid only in the FILE SOURCEOUT clause. The clause allows the maximum size of the output file to be defined. The default is 5000 records. A FILE.SIZE of greater than 5000 will be replaced by 5000. A warning is issued if the default is used.

The message clause allows certain control options to be passed to the RPG compiler. Any options supplied in this way will override any corresponding control options in [PATCHFILE] or [SOURCEIN]. The < text > is a list of the options, each option introduced by a dollar sign (\$) and terminated by a space, the dollar sign of the next option, or the end of the message. Spaces are otherwise ignored. The following are the dollar options which may be specified

LIST
 LOGIC
 SEVERE
 SUPR
 XMAP
 MAP
 NAMES
 NEW
 MERGE
 SEQ (resequence will start at 00100 by increments of + 10)
 XREF

A full description of these options is to be found in the CMS RPG Reference Manual, form number 2007274.

An option may be reset by inserting the character "N" immediately before the name of the option, for example \$NSEVERE.

Although any option specified in < text > will override any corresponding option in [PATCHFILE] or [SOURCEIN], it will not replace any such option in a [SOURCEOUT] file created by the compilation. The < text > options alter the performance of this compilation only—a subsequent compilation from [SOURCEOUT] will be unaffected.

Example CO RPG SY FI SOURCEIN FID MASTER
 FI PATCHFILE FID PATCHES FI SOURCEOUT
 FID NEWMAST RECORD 80 RECORDS.BLOCK 9
 FILE.SIZE 50 MESSAGE \$NEW \$MERGE \$NLIST

If PATCHES contains \$LIST, NEWMAST will contain \$LIST although no listing will be produced by the example compilation. NEWMAST will not contain \$NEW or \$MERGE.

The PSSIZE clause allows the perform stack size of an RPG program to be defined. The < integer > may optionally be followed by the character "K" to denote multiples of 1024. The evaluated result of < integer > (including modification by K) must not exceed 65535. The default value is 10.

The LIBRARY or SYNTAX clause determines whether the compiler is to generate an object program code-file or not. LI or LIBRARY will cause the compiler to build an object-program file of name < object—program—name > on the disc identified by < disc—id—2 > (or the system disc if < disc—id—2 > is omitted). The appropriate language reference manual defines the default name used if none is specified. The object—program will only be output if no syntax errors were discovered in the source program. SY or SYNTAX will cause the compiler to check the source for correct syntax only, and no object—program will be generated.

The utility provides extensive macro—call facilities as follows:

The construct _____ * _____ < disc—id > / _____ < file—id > _____

may be used to replace any part or parts of the input message between a circled A and any following circled B.

Examples

1) EX ABC/CO *ABC/COMPILE

Where the file COMPILE on disc ABC contains a full specification string

2) CO ABC/OBJECT *DEF/COMPOPTION FI PATCHFILE FID SOURCE
Where the file COMPOPTION on disc DEF contains DEF/COBOL LI PRINT

3) CO OBJECT RPG LI *SOURCE *NEW MESSAGE \$NEW
where the file SOURCE on the system disc contains FI PATCHFILE FID PATCHES FI SOURCEIN FID SOURCE DEVICE CASSETTE and the file NEW on the system disc contains FI SOURCEOUT FID CLEANSO RECORD 80 RECORDS.BLOCK 2 DEVICE CASSETTE FIL.SIZE 400

Output Messages

The following messages will be output by the utility in response to incomplete specifications or in the event of the appropriate error.

Warnings

- 1) KEYWORD ONLY VALID FOR FILE MODIFICATION—IGNORED
A keyword used for file modification has been found outside a file modification clause.
- 2) WARNING—PROG—ID TRUNCATED TO 12 CHARS
A program—id of greater than 12 characters has been found. Only the leftmost 12 characters are used.
- 3) WARNING—PROG—PACK—ID TRUNCATED TO 7 CHARS
A program—pack—id of greater than 7 characters has been found. Only the leftmost 7 characters are used.
- 4) INVALID VALUE IN STACK CLAUSE—DEFAULT USED
An invalid value was found in the PSSIZE clause. The default value of 10 is taken.
- 5) WARNING—THE FILE ATTRIBUTES OF SOURCEOUT WERE INCOMPLETE
An attribute RECORD, RECORDS.BLOCK or FILE.SIZE was not specified.

Errors

The following errors will prevent the ZIPing of the compiler.

- 1) ILLEGAL PARAMETER LIST—JOB TERMINATED
An * has been found with no immediately following < file—name > .
- 2) < file—name > —FILE NOT FOUND
The file specified in an * < file—name > construct is not on line.
- 3) KEYWORD TOO LARGE IN INPUT
A keyword greater than 32 characters has been found. This may occur as the result of a previous error.
- 4) INVALID KEYWORD IN INPUT—< text >
An expected keyword was not found. < text > identifies the characters found where the keyword was expected.
- 5) INVALID KEYWORD IN FILE MOD
An expected keyword in the file modification was not found. The next word is skipped on the assumption that the keyword was mistyped.
- 6) INVALID ATTRIBUTE FOR OLD FILE
The keywords RECORD, RECORDS.BLOCK, and FILE.SIZE are only valid for the file SOURCEOUT.
- 7) INVALID DEVICE TYPE < text >
A non—CMS device identified by < text > has been found in a file modification.
- 8) NO COMPILER GIVEN BEFORE FILE—JOB TERMINATED
No compiler was specified before the first file modification.
- 9) INVALID CALL OF *FILE—NAME — JOB TERMINATED
A * < file—name > construct has been found in an illegal location.

10) NESTED MACRO CALL FOUND—JOB TERMINATED

An * < file—name > construct has been found within the expansion of another similar construct.

11) NO SPECIFICATIONS GIVEN—JOB TERMINATED

The initiating message was empty.

12) NO COMPILER SPECIFIED—JOB TERMINATED

No specific compiler was identified in the initiating message.

13) ERROR WHEN EXPECTING \$

The message clause contains an option not immediately preceded by a dollar sign.

14) NO KEYWORD FOUND AFTER \$

The MESSAGE clause contains a dollar sign with no immediately following option.

15) INVALID KEYWORD FOUND AFTER DOLLAR

A dollar option not in the permitted list has been specified in the MESSAGE clause. The description of the MESSAGE clause earlier in this section contains a full list of the permitted options.

16) ZIP FAILURE— < reason >

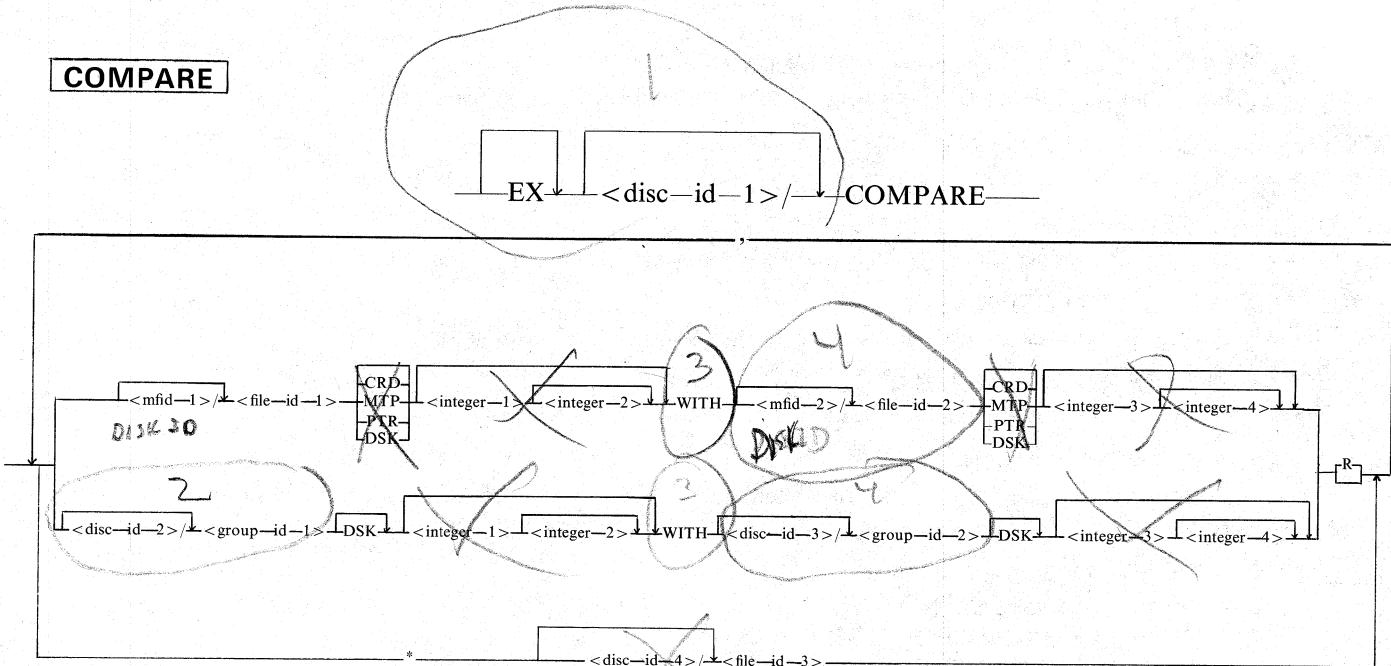
< reason > may be PROGRAM NOT FOUND
INTERPRETER NOT FOUND
NO MEMORY AVAILABLE
NO DISK AVAILABLE
MIX IS FULL

Execution of the specified compiler did not commence.

The cause of the failure is identified by < reason > .

Note that all the above messages are output by the utility. Any applicable system message may be output by the MCP, (for example DUPLICATE FILE).

COMPARE



This utility allows a list of files (or groups of files) to be compared with other files (or groups of files). The utility may be executed from a disc other than the system disc by specifying `< disc-id-1 >`. A line printer or equivalent (for example, console printer) is required to use this utility.

The utility will accept a list, separated by commas, of comparison specifications. Each comparison specification consists of two file specifications separated by the word "WITH", and followed by the optional realignment character, R. Any or all trailing comparison specifications may be input to the utility via the file identified by `< file-id-3 >`, which is assumed to be on the system disc unless `< disc-id-4 >` is specified.

Semantics

When the device type is CRD, MTP, or PTR, or is DSK and the group-id construct is not used, then the following semantics apply.

The `< mfid-1 >` and `< mfid-2 >` entries may only be used for disc and magnetic tape (including cassette) files. If the `< mfid >` field is left blank, then, for disc the system disc is assumed, and for magnetic tape a single file tape is assumed.

The files to be compared are identified by `< file-id-1 >` and `< file-id-2 >`. The files are assumed to be on disc unless the associated device option, CRD, MTP, PTR, is used. CRD defines any 80 column or 96 column punched card device. MTP indicates that the corresponding file is on magnetic tape or cassette. A paper tape input file may be defined by PTR. The DSK entry may be used for documentary purposes, since the default device type is disc.

The record size in bytes of the files to be compared may be defined using `< integer-1 >` and `< integer-3 >`. The blocking factor (number of records per block) may be defined using `< integer-2 >` and `< integer-4 >`. The record size of files being compared must not exceed 1024 bytes. If the record size of a file does exceed this figure, then, in most cases, the blocking factor option may be used to enable comparison of the files. For example, a file with record size 4160 can be reblocked as 104 blocked 40. If the card file is a large prime number P, the file may be compared using a record size of 1 and blocking factor of P, but this method will be very slow.

If the records to be compared are of different lengths, only the number of characters corresponding to the shorter record are compared.

If the realignment option (R) is specified, the following algorithm is used in an attempt to realign the records being compared:

If three consecutive records fail to compare, then the utility will attempt to compare the 3rd of these records from < file—id—2 > with the next two records from < file—id—1 >.

If all five comparisons fail, then the utility will attempt to compare the 5th non-comparing record from < file—id—1 > with the 4th, 5th, 6th, and 7th from < file—id—2 >.

If realignment fails, then the utility will discontinue the comparison, and proceed to the next comparison required by the input specifications.

If a correct comparison occurs at any stage, the normal mode of processing restarts.

The following additional semantics apply to disc files when the group—id construct is used.

Membership of the group of files to be compared is determined by < group—id—1 >. The non-group—id—1 portion of each related file is concatenated with < group—id—2 > to generate the name of the file with which it is to be compared. The groups are assumed to be on the system disc unless the associated < disc—id > is specified.

Examples

DISK 1 contains files A, B, C, D, AB, AC, ABC, BC
DISK 2 contains files A, B, C, D, AB, AC, ABC, BC, BD, EF
(System Disc)

COMPARE DISK 1/= WITH =
will compare all files on DISK 1 with the corresponding files on DISK 2.

COMPARE = WITH DISK 1/ =
will compare all files on DISK 2 with the corresponding files on DISK 1. It will fail to find DISK 1/BD and DISK 1/EF.

COMPARE DISK 1/A= WITH A=
will compare files A, AB, AC, and ABC on DISK 1 with the corresponding files on DISK 2.

COMPARE DISK 1/A= WITH AB=
will compare

DISK 1/A with Disk 2/AB
DISK 1/AB with DISK 2/ABB*
DISK 1/AC with DISK 2/ABC
DISK 1/ABC with DISK 2/ABBC*

* Note that a "file not found" condition will result for these cases.

COMPARE A with=
will compare A with itself.

The following messages may be output by the utility to the printer in the event of the corresponding error or condition.

- 1) CANNOT OPEN FILE—< file—id >
This message is output if a file cannot be opened—usually because it is not present.
- 2) END OF FILE < file—id—1 > BEFORE END OF FILE < file—id—2 >
This message is output if the end of one file is detected before the end of the other.
- 3) < file—id—1 > WITH < file—id—2 > COMPARISON COMPLETE < integer > ERRORS
This message is output if the ends of both files have been reached together. The number of errors is indicated by < integers >.

4) ILLEGAL SYNTAX FOR ITEM—< input string >

This message is output if an item in the comma list is not syntactically correct. The utility will attempt to follow the message with a character string which identifies the area in error.

5) FAILED TO REALIGN—COMPARISON TERMINATED

This message is output if realignment has been specified but has failed.

6) DIFFERENCE DETECTED AT BYTE n

This message is output if two records fail to compare. This is followed by

< filename 1 > RECORD n IS:

< record—print >

< filename 2 > RECORD m IS:

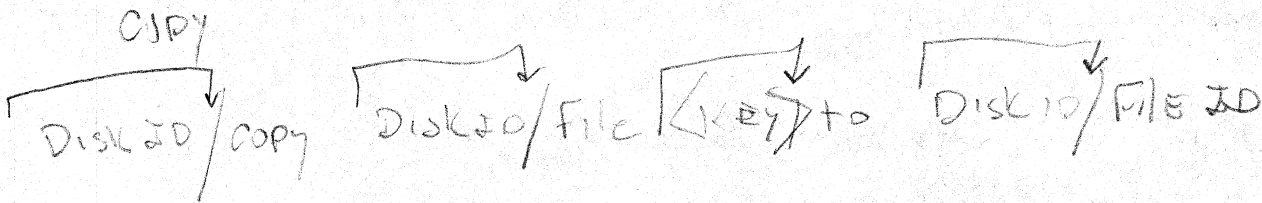
< record—print >

where < record—print > has the format:

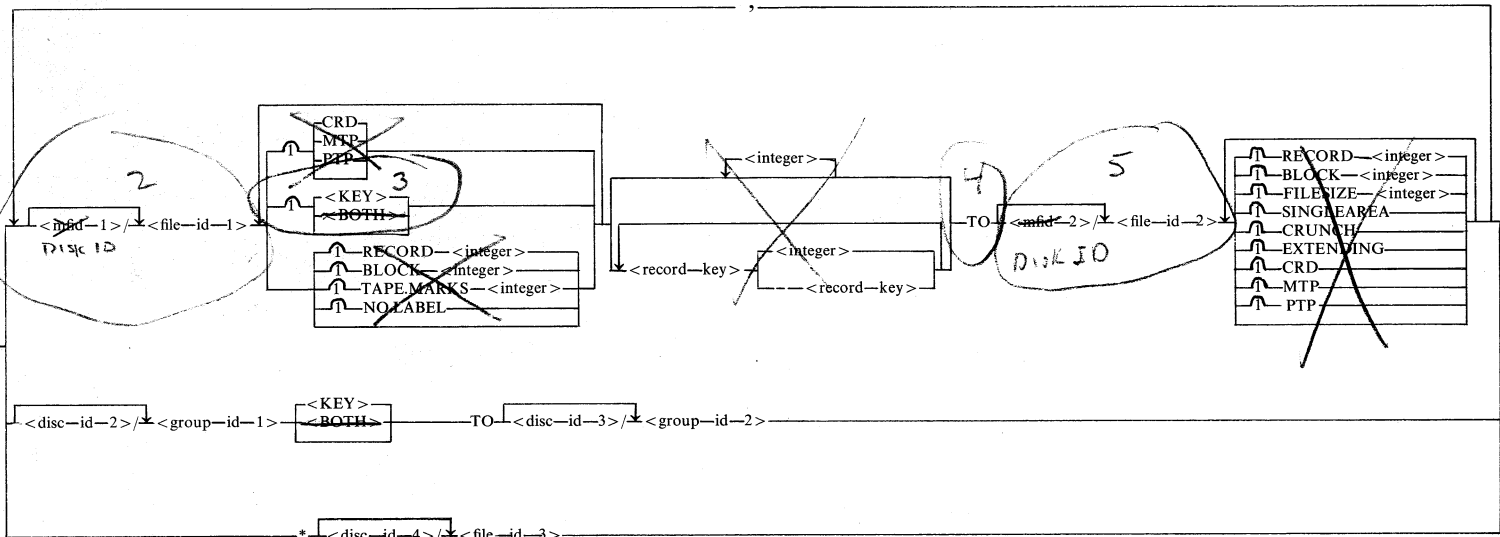
< byte—offset > < 32—bytes—in—hex > “< 32—bytes—in—ASCII >”

and .. is used to represent a 00 character in < 32—bytes—in—hex > and a blank is used to represent a non-printable character in < 32—bytes—in—ASCII >

Any applicable system message may be displayed on the system journal by the MCP (for example, NO FILE if no printer is available).



EX → ^① <disc-id-1> - / - COPY



This utility provides for the copying of files, either in whole or in part, from any CMS device to any other CMS device. The utility allows the names and attributes of the copies to be specified, and also permits groups of files to be copied on disc. A list of copy requests, separated by commas, may be presented to the utility, and each request will be processed in sequence. The failure of a particular request will not necessarily prevent compliance with subsequent requests.

The utility may be executed from the system disc, or any other disc if < disc-id-1 > is specified.

Specific File Copy

The file to be copied is identified by < file-id-1 >. The name of the output file is determined by < file-id-2 > (and will always be a new file except when the EXTENDING option is used as described below). When either file resides or is to reside on magnetic media, the associated < mfid > may be used to specify the disc-id or multifile identifier. If an < mfid > is omitted, then, for disc, the system disc is assumed, and for magnetic tape (and cassette), a single file tape is assumed.

The utility assumes that all files reside or are to reside on disc unless the relevant device identifier is used. CRD refers to punched card files, MTP refers to magnetic tape (and cassette) files, and PTP refers to paper tape files.

Attributes of the input and output files may be set as follows:

Record and Block Size

The RECORD and BLOCK options may be used with both input and output files.

The number of bytes in the record or block is specified using the corresponding < integers >. The record and block sizes of input disc files are always taken from the Disc File Header. The record and block sizes of non-disc input files are determined in the following order:

- RECORDRECORD < integers > then < device-default >
 - BLOCK < integers > then RECORD < integer > then < device-default >
- where < device-default > is

- 180 characters (PTP and unlabelled MTP)
- Device Capability (CRD—either 80 or 96 characters)
- Label Value of Attribute (labelled MTP)

Note that the utility will be unable to copy labelled MTP files whose record/block sizes are greater than 1024 characters unless the record and block sizes are specified in the input specifications.

The record and block sizes of output files will be identical to the defined or defaulted in put file values unless the record or block options are used. If the output record size is less than the input record size, then bytes will be lost from the right of the record. If the output record size is greater than the input record size, the additional bytes will be filled with ASCII spaces (if the file type is SOURCE or DATA) or binary zeroes (for any other file type).

Note: the utility will check that the record and block values to be used are consistent (that is, the block size must be an integer multiple of the record size).

Care should therefore be taken when only specifying block-size, since the default record size may not be compatible.

Example. A labelled magnetic tape contains nine 8 character records per block. If the record size is not specified and the block size is specified as 36, then the BAD ATTRIBUTE message will be output when the utility opens the file.

Filesize

The FILESIZE < integer > option may be used to declare the maximum size for output files.

Singlearea

The SINGLEAREA option may be used to specify that an output disc file is to occupy a single disc area. SINGLE-AREA is automatically set when the file being copied is of filetype SYSTEM or CODE.

Crunch

If the CRUNCH option is used, the output disc file will occupy the minimum area of disc, but can never be extended. CRUNCH is automatically used when the file being copied is of filetype SYSTEM or CODE.

Extending

The EXTENDING option allows records to be added to the end of an existing disc file. The existing file must have identical attributes to the file being copied.

Unlabelled Magnetic Tapes

The NO.LABEL option is provided to permit the COPYING of unlabelled (or labelled non-CMS) files. The < file-id-1 > entry must be used in order to identify the file for SCL purposes. MCP will output the message "< mx > /COPY < 14 > WAITING UNLAB SPURIUS/ < file-id-1 > MT DEVICE REQUIRED".

An "AD < mx > /COPY < tape-peripheral > message must be used to identify the unlabelled file. The end of file recognition for unlabelled files is determined by tapemark count as described below.

The TAPE.MARKS option allows the user to specify the aggregate number of tape marks which will indicate end of file to the utility when copying an unlabelled file. The default value is 2. Each tape mark which is encountered will attribute to this total, thus a standard labelled CMS file will be copied up to, but excluding, the trailing label if NO.LABEL and two tape marks are specified. (A labelled CMS file consists of "Label TM data TM label"). The user must therefore be aware of the format of any file which is to be copied when using the NO.LABEL option.

Disc Indexed Files

The utility assumes that a copy of the associated data file is required (in key order) when the file identified by < file-id-1 > is recognised as a key file. If a copy of the key file only is required, the < KEY > option may be used. In either case, the output file is identified by < file-id-2 >. If a copy of both the data file and the key file is required, the < BOTH > option may be used. The output key file will be identified by < file-id-2 >, and the output data file will be identified by the concatenation of < file-id-2 > and QQ. The output key file parameter

block will be altered to refer to the output data file. When copying a data file via a key file, the < record—key > options of the selective copy specification may be used as described below. Note that selective copying cannot be performed if < BOTH > is specified.

Selective Copy

The capability is provided to copy only selected portions of an input file via the selective copy specification. The various formats of selective copy specification apply to disc and non-disc files as follows.

Non-disc files—The < integer—list > may be specified for paper tape (PTP), punched card (CRD), magnetic tape and cassette (MTP). Each < integer > is a decimal number of up to 7 digits, and each pair of < integers > in the string specify a starting record number and the number of records required from that point respectively, Relative record numbers commence with record 1. If there is an odd number of < integers >, then the last < integer > is equivalent to a pair of < integers > which would cause COPY to process to the end of the file. The < integer—list > must not specify any random access of the file, that is, the first < integer > of each < integer > pair must identify a higher record number than any previously copied record. The < record—key > option is not applicable to non-disc files.

Disc files—The < integer—list > applies to all disc files and has identical use to that described above for non-disc devices, except that random access is permitted when using disc. The < record—key > option allows selective copying of an *indexed* file by reference to the key value contained in each record. The user may specify a number of records to be copied via the < integer > option, or a terminating key value via the < record—key > option. The copy will include the records containing the starting and terminating values of < record—key >. Selective copying cannot be performed when < BOTH > is specified.

Disc Group Copy

The group to be copied is identified by < group—id—1 > and is assumed to be on the system disc unless < disc—id—2 > is specified. Each file which is found to be a member of the group will be copied to the disc identified by < disc—id—3 > (or the system disc if < disc—id—3 > is omitted), and the non-< group—id—1 > portion of the file—id will be concatenated with < group—id—2 >.

Indexed Files

If any file in the input group is recognised as a key file, then a copy of the data file associated with the key file will be produced (in key order) unless the < KEY > or < BOTH > option is used. The file—id of the output data file will be developed as described above. If < KEY > is specified, then, when any file is recognised as a key file, only the key file will be copied, and the output file—id will be developed as described above.

If < BOTH > is specified, then, when any file is recognised as a key file, both the key file and the associated data file will be copied. The file—id of the output key file will be developed as described above. The file—id of the output data file will be the file—id of the output key file concatenated with QQ. The key file parameter block of the output key file will be altered to refer to the output data file.

Output Messages

The following messages are printed after a request has been processed. The utility will then continue with the next request, or terminate if the list of requests is exhausted.

- 1) NO FILES FOUND IN THE FAMILY < group—id >
No file whose name contains the leading characters < group—id > was found on the specified disc.
- 2) DISK < disc—id > NOT AVAILABLE
The disc identified by < disc—id > is not ready or not installed.
- 3) < file—name > NOT FOUND
No file of name < file—name > was found on the specified disc. If the specified file contained the input parameters, the utility will terminate.

- 4) < integer > , < integer > IN < file—name > NOT COPIED
If a record number in an integer pair indicates a section of the file at a lower file address than a previously specified section then the pair is ignored by the utility and the above message is output.
- 5) < file—name > EXHAUSTED DURING < integer > , < integer >
If end-of-file is encountered while the section of the file indicated is being copied, then the above message is displayed.
- 6) < file—name > TO < file—name > BAD ATTRIBUTES
If a particular attribute specification is either meaningless or inconsistent, then this message will be displayed. The inconsistencies will be in the relationship between output device, record size, and block size.
- 7) FILE IDENTIFIER < file—name > TOO LONG
If a copy of a group of files is requested and a < file—name > generated by the utility is longer than twelve characters, then the above message is output, and the file is not copied.
- 8) < file—name > TO < file—name > COPY DISCREPANCY
This message is output if end-of-file has been reached before it is expected, and implies erroneous information in the Disk File Header.
- 9) < file—name > TO < file—name > COPIED
This message will be displayed at the conclusion of each successful file copy.
- 10) < file—name > NOT COPIED—ILLEGAL REQUEST
If, when copying a < group—id > to another < group—id > an individual file—name is generated which is "SYSMEM" (a name reserved for system use) or all spaces, the above message is displayed, and the file will not be copied.
- 11) < file—name > NOT ACCEPTABLE—RECORD SIZE OF m IS GREATER THEN THE MAXIMUM SPECIFIED FOR THIS RUN—RESUBMIT
This message will be displayed if a file is submitted to the utility with a record size greater than that expected. This can happen if an MTP file with a record size greater than 1024 characters is submitted to the utility without the record size being properly specified in the parameters. This request should be respecified, since this invocation of the utility will ignore it and continue with the next request, if any.
- 12) < file—name > EXHAUSTED DURING RANGE < record—key > < integer >
This message is displayed if end of file is encountered while the section of the file indicated by < record—key > < integer > is being copied.
- 13) < file—name > EXHAUSTED DURING RANGE < record—key > — < record—key >
This message is displayed if no records were found in the range < record—key > < record—key > .
- 14) NO RECORDS FOR COPYING FROM < file—name >
This message is displayed if no records were found for copying in the specified file.
- 15) INPUT RECORD < integer > OUTPUT RECORD < integer >
This message is displayed if an irrecoverable error is encountered. One of the following messages will be displayed to identify the error
 - PERMANENT ERROR ON INPUT FILE
 - PERMANENT ERROR ON OUTPUT FILE
 - OUTPUT FILE TOO SMALL

The request will be aborted, and message 16) below will be displayed.

- 16) < file—name > TO < file—name > COPY FAILURE
This message will be displayed for each file copy which has failed for some previously specified reason.
- 17) NO RECORDS IN THE KEY FILE
This message is displayed if, during a copy of < BOTH > files, the utility was denied access to a data file

through some failure in the key file.

- 18) < file—names > NOT FOUND FOR EXTENDING
This message will be displayed if a file requested for extension is not found.
- 19) < file—name > REMOVED
This message is displayed when a previously existing file of the same < file—id > is removed to permit closing the new file.
- 20) < file—name > TO < file—name >
This message will be followed by one of

**SELECTION CRITERIA IGNORED
EXTENDING FLAG IGNORED**

When parameters are input which would require re-creating a key file.

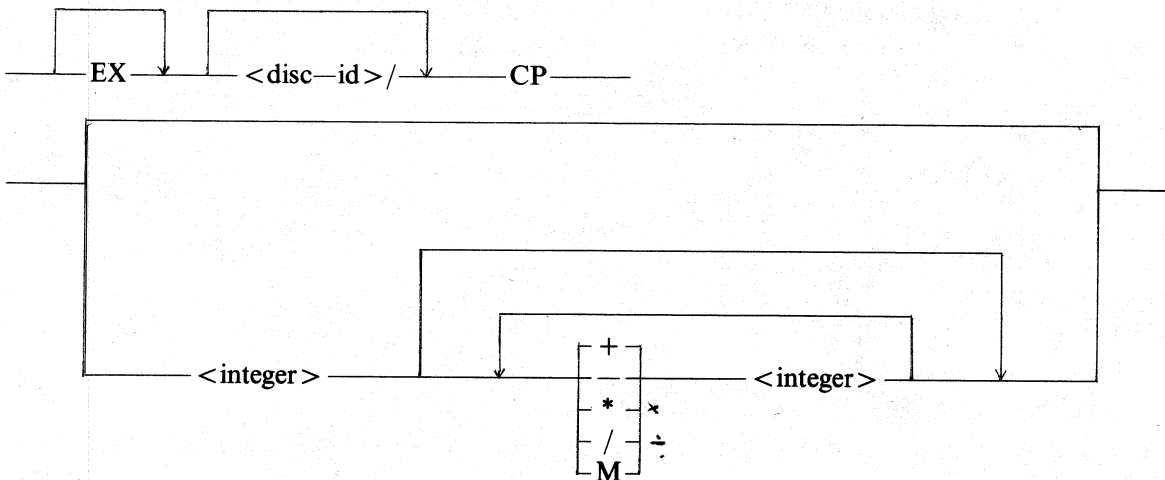
The following messages will be output, and the utility will terminate, if errors are found in the initiating message.

- 1) ILLEGAL PARAMETER LIST
The utility will attempt to follow this message with a number of characters from the area of the input message which caused the failure.
- 2) INVALID CHARACTER IN IDENTIFIER < identifier >
This message is displayed if the < disk—id > or < file—id > portions of the file—name contain characters which are illegal in file—names.
- 3) BAD ATTRIBUTES SPECIFIED
This message will be displayed if inconsistent attributes are specified for an input file. The utility will then display ILLEGAL PARAMETER LIST and terminate.

Note that all the above messages are generated by the utility. Any applicable system message may be output by the MCP in addition to the above (for example, NO FILE).

CP

CP (Compute)



This utility displays the calculated value of `< compute-string >` (an input calculation specification). The utility may be executed from a disc other than the system disc by specifying `< disc-id >`.

The utility will return the value of `< compute-string >` in both hexadecimal and decimal formats, the calculation being performed on a strictly left to right basis, that is, parentheses etc. are not allowed.

The `< integers >` accepted by the utility are any hexadecimal or decimal numbers in the range

-999 999 999 999 999 [- @ 38D7EA4C67FFF@]
to
+999 999 999 999 999 [@38D7EA4C67FFF@].

The operators `+`, `-`, `*`, `/`, `M` provide addition, subtraction, multiplication, division, and modulus division (the result is the remainder) respectively.

If the `< computer-string >` is provided in the initial parameters, then the utility will terminate after computing the value.

If the `< compute-string >` is omitted from the initial parameters, then the utility will issue ACCEPTS through which subsequent `< compute-strings >`s may be input, for example `AX < mix-number > < compute-string >`. The utility will continue to execute until an AX is input with no `< compute-string >`.

Output Format

The computed value of `< compute-string >` will be output in the form

CP: `< hexadecimal-value > = < decimal-value >`

Error Messages

The following error messages may be displayed by the utility. If the utility is in the ACCEPT mode of operation, the utility will not terminate.

1) CP: NUMBER TOO LARGE

This message is displayed if at least one number in `< compute-string >` is out of range.

2) CP: MISSING OPERAND

This message is displayed if two consecutive operators, or an illegal symbol, is detected in `< compute-string >`.

3) CP: HEX. NO. WITH MISSING "@"

This message is displayed if an "@" sign is omitted, or if an illegal symbol is found in `< compute-string >`.

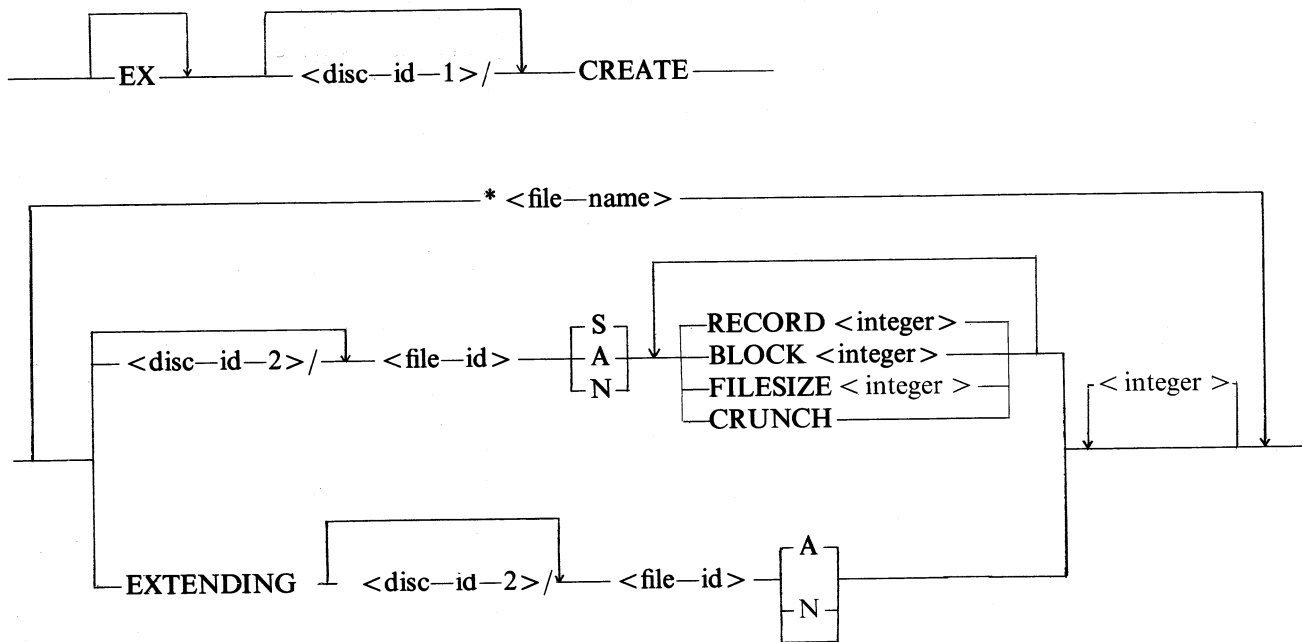
4) CP: INVALID OPERATOR

This message is displayed if an operator is omitted, or an invalid symbol appears in `< compute-string >`.

5) CP: OVERFLOW

This message is displayed if an intermediate value became out of range in the course of the calculation.

6) CP: DIVISION BY ZERO



This utility will create or extend disc-files using data input through the console keyboard. The utility may only be used with systems incorporating a console keyboard. The utility can be executed from a disc other than the system disc by specifying < disc-id-1 >.

The file to be created or extended is identified by < file-id >. The file is assumed to be on, or will be output to, the system disc unless < disc-id-2 > is specified. The utility may only be used with files of type Source or Data.

When the utility is used to create a new file, certain attributes may be defined. These attributes will be taken from the old file when the extend mode is used. In the create mode, S specifies a source file, and the keyboard input will be in alphanumeric format. A or N specifies a Data file with keyboard input in alphanumeric or numeric (hexadecimal) format respectively. The default filetype is S. In the extend mode the filetype is taken from the old file, and A or N defines the type of keyboard input as above. The default is A. Alphanumeric input is accepted as direct keyboard input, whereas numeric input will require two characters (0-9, A-F) for each byte of the record.

The RECORD clause allows the number of characters per record of a new file to be defined. If no record size is specified then the utility assumes a record size of 80 bytes for source files, and 180 bytes for data files.

The BLOCK clause allows the number of characters per block of a new file to be defined. If a record size is specified but no block size is specified then the block size is taken to be equal to the record size. If neither record size nor block size is specified, then the record sizes as defined in the RECORD clause, above, are assumed, and the block size is taken to be 160 bytes for source files and 180 bytes for data files.

The FILESIZE clause allows the maximum number of records likely to be written to the new file to be specified. This facility may be usefully employed to allocate only as much disc space as is required. The default is 32,768.

The CRUNCH clause allows the new file to be closed with the crunch flag set. The new file will occupy the minimum area of disc, but can never be extended.

The integer list specified in the syntax may be used to provide "tab" positions within the record. The use of OCK1 when keying input data causes the utility to reposition the input point in the record to the next tab position. During this repositioning, the utility will fill all character positions left unspecified in the record with a fill character determined

by the input type. For source input, the fill character will be an ASCII space, for alphanumeric input an ASCII zero, and for numeric input a binary-zero filled character. The record length plus one is used as a terminating tab position (whether or not other tab positions are specified).

The utility can be used for record sizes up to 500 bytes, but, since the utility cannot be given input greater than the width of the console, tab position are mandatory on files of larger record sizes. For example, a file of 180 byte records requiring alphanumeric input will require at least one tab position (for instance at position 100), whereas a file of 180 byte records requiring hexadecimal input will require a minimum of two tab positions (for instance at positions 60 and 120).

The utility operates in two modes; Record Input Mode (entered from PK1) or Record Modify Mode (entered from PK2). The PK associated with the currently active mode is disabled (the light is turned off) to indicate the appropriate mode. PK6 is enabled at appropriate points in either mode to close the file and terminate the utility. When execution of the utility begins, Record Input Mode is automatically entered.

PK1 Record Input Mode

In Record Input Mode, the utility will print a sequential record number and then allow the operator to input characters until the record is complete. OCK1 will terminate each line of input as either a whole record is completed or input is complete up to the next relevant tab position. Where possible, echo-typing during record input will show the relevant position of the input characters in the record. When the record is completed the utility will illuminate all other usable PK options for possible selection.

PK2 Record Modify Mode

The point in the record at which alterations are to be made is selected by presenting an identifying character string which immediately precedes the required byte of the record. The character string for insertion or replacement follows the identifying string, delimited by colons (:). If alterations are to be made at the beginning of the record, no identifying string is input. The syntax of the keyboard input is:—

| < identifying—string > ↓ : < modifying—string > :

The use of OCK1 or OCK2 to terminate the input determines whether the < modifying—string > will replace or be added to the existing characters in the record.

OCK1—Replacement

The < modifying—string > replaces the corresponding number of characters in the record if OCK1 is used to terminate the input. For example with a record containing

ABCD0123

the amendment C:XY: would result in a record containing

ABCXY123.

OCK2—Insertion

The < modifying—string > is inserted at the indicated point of OCK2 is used to terminate the record. The insertion can cause characters in the record to be moved to the right. The shifting of characters applies only to those characters from the starting byte to the next higher relevant tab position; characters beyond this tab position will not be affected.

For example, with a record containing

ABCDEF~~G~~H1234

The amendment C:WXYZ: would result in the record
ABCWXYZD1234 if the tabs specified were 7 and 9, or the record
ABCWXYZDEFGH if the tab specified was 7 alone.

On completion of a modification, the utility will print the amended record with its associated record number, and then illuminate all other usable PK options for possible selection.

PK6—Terminate

PK6 will take the utility to end-of-job when all desired keyboard input has been made.

Note that the EXTENDING facility allows the creation of large files to be performed in more than one run of the utility.

Output Messages

The following messages will be output by the utility in the event of the appropriate error.

Errors in Initial Parameters

These errors occur if the initial parameters following "CREATE" (or "EXTENDING") in the SCL input string are incorrect. After the message is displayed, the utility terminates.

- 1) FILETYPE IS NOT SOURCE OR DATA
This message is displayed if the file identified for extension is of an incorrect file—type.
- 2) < file—name > NOT FOUND
This message is displayed if the file identified for extension or the parameter—file identified by the * < file—name > macro-call cannot be located.
- 3) ILLEGAL PARAMETER LIST—ATTRIBUTE SPECIFICATION INVALID
This message is displayed if there is an incompatibility between the specified record and block sizes.
- 4) ILLEGAL PARAMETER LIST—TABS ERROR
This message is displayed if tab positions beyond the end of the record are specified, or imply input fields larger than the capability of the console.
- 5) ILLEGAL PARAMETER LIST
The utility will attempt to follow this message with a number of characters from the area of the input message which caused the failure.
- 6) INVALID CHARACTER IN IDENTIFIER < identifier >
This message is displayed if the < disk—id > or < file—id > portions of the filename contain characters which are illegal in file—names.
- 7) NO SPECIFICATION GIVEN
This message is displayed if no parameters are entered following CREATE in the SCL input string.

Errors During Execution

The following errors may be encountered during execution of the utility, and do not cause the utility to terminate.

The messages are displayed via the console file.

1) NOT HEXADECIMAL CHARACTER INPUT—RESUBMIT

If the input mode is hexadecimal, any character other than 0—9 and A—F will cause the above message to be displayed. The complete entry must be resubmitted.

2) ODD NUMBER OF HEXADECIMAL CHARACTERS INPUT

If the input mode is hexadecimal, and the number of hexadecimal characters is odd, then this warning is displayed. The utility processes the hexadecimal string by extending it on the right with a zero character.

3) INPUT ERROR—RESUBMIT RECORD MODIFICATION

If the syntax of the keyboard input in Record Modify Mode is incorrect, the above message is displayed. The entire entry is discarded.

4) BYTE WITHIN RECORD SPECIFIED NOT FOUND

The < identifying—string > in the keyboard input in Record Modify Mode could not be found in the record. The entire entry is discarded.

5) UNWANTED KEY PRESSED—PLEASE RE-INPUT

This message is displayed if an unexpected terminating key is used. The entry should be re-submitted.

6) INPUT IMMEDIATELY BEFORE PK6 HAS BEEN LOST

This warning message is displayed if characters were input immediately before PK6 was pressed to terminate the utility. The characters will not be written to the output file, and the utility will terminate normally.

Note that all the above messages are generated by the utility. Any applicable system message may be output by the MCP in addition to the above (for example, DUPLICATE FILE).

DP (Discontinue and Dump)

DP

———— DP ———— < mix # > — / — < prog—id > ————

This function, implemented as an intrinsic within the MCP, has the same effect as a DS command with the exception that the virtual memory space allocated on the disk is updated to include the complete run structure of the task and the whole is closed as a file and locked. The name of the file is DMFILE n, where n is the mix number of the dumped program. This may be analyzed by a dump analyze program. All the restrictions of the DS command apply and the message

< mix # > / < prog—id > DP'ED

is displayed on successful completion of the function.

An invalid DP request is rejected, and the invalid DP message is returned with an added rejection clause as follows:

< invalid—request > INVALID

is displayed if the mix—number does not correspond to an executing task, the program—id does not correspond to the mix—number, the target task is an MCS, or the request is not correctly formatted.

< invalid—request > INVALID—NEEDS PROGRAM ID.

is displayed if the mix—number alone has been specified.

DS

DS (Discontinue Program)

_____ DS _____ < mix # > _____ / _____ < prog-id > _____

This function, an intrinsic within the MCP, causes the orderly termination of the specified task. All resources used by the task are released to the MCP for reallocation and all files opened by the task are closed. The disk space allocated for the virtual memory of the task is deallocated.

The MCP checks that the mix number and program name quoted in the command match. Otherwise, the command is ignored and the invalid DS message is displayed. A data communication Message Control System (MCS) cannot be the object of a DS command and the invalid DS message will be displayed if this is attempted.

Upon successful completion of the DS, the message

< mix # > / < prog-name > DS'ED

is displayed. If the DS is invalid then the request is rejected and the message is returned with an added rejection clause as follows:

< invalid-request > INVALID

is displayed if the mix-number does not correspond to an executing task, the program-id does not correspond to the mix-number, the target task is an MCS, or the request is incorrectly formatted

< invalid-request > INVALID-NEEDS PROGRAM ID

is displayed if the mix-number alone has been specified.



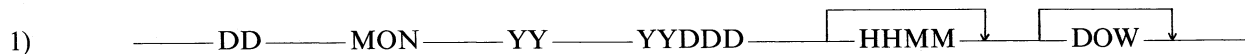
This intrinsic allows the system date and time to be interrogated or changed.

If DT is used alone, the current system date (and time if the system contains a real-time clock) will be displayed.

If date and time are both input they must be in that order. The date field is eight bytes in size and the time field is four bytes. The <MM>, <DD>, and <YY> fields are one or two digit fields which will be range checked for respective validity (for example, an <MM> entry of 0, or greater than 12 will be rejected). The <HHMM> field must be a four digit entry not greater than 2359. Bad data in "date" or "time" will not result in rejection of the entire input. Any remaining or prior field with good data will be accepted.

Output Messages

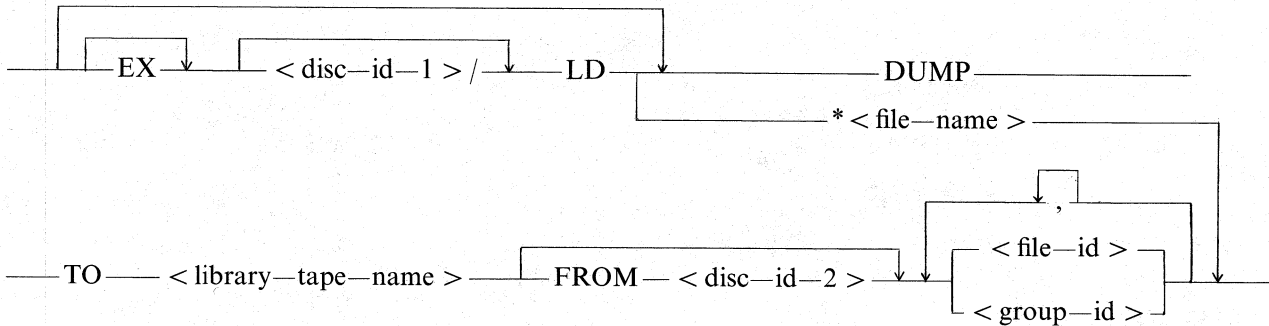
The system will display the following messages in response to the DT command.



- 1) This message is displayed at the end of the DT command to show the current (or new) value of the system date and time. DD represents the day of the month, MON is a three character abbreviation of the month, YY represents the least significant digits of the year. YYDDD shows the date in the Julian form. The time entry HHMM will only be displayed by systems containing a real-time clock. The DOW entry may be output by some CMS implementations, and consists of a three character abbreviation of the day of the week.
- 2) < INVALID >
This message is displayed if the field size of either date or time is incorrect. The entire input is rejected.
- 3) < INVALID DATE >
This message is displayed if any of the fields <MM>, <DD>, or <YY> are outside the appropriate range. The entire date is rejected, but a valid time entry in the same message will be accepted.
- 4) < INVALID TIME >
This message is displayed if a time entry of greater than 2359 is detected. The time will be rejected, but a valid date entry in the same message will be accepted.
- 5) < NO CLOCK >
This message is displayed if a time entry is input to a system with no real-time clock. A valid date entry in the same message will be accepted.

DUMP

DUMP (Dump Files to Library Tape)



Note that this function is a sub-program within the utility LD. MCP recognises the mnemonic DUMP if LD is not specified, and will automatically initiate a load of the LD utility. The < file-id > of LD should not therefore be changed if the DUMP function is to be invoked in this manner. To discontinue the DUMP function, "DS < mix-number / LD" must be used.

This function may be executed from a disc other than the system disc by specifying LD and < disc-id-1 >.

The function will create a library tape of name < library-tape-name > containing files copied from the disc specified by < disc-id-2 > (or the system disc if < disc-id-2 > is omitted).

The contents of the disc will not be altered. The file list identifies the particular files or groups of files to be copied to the library tape. Note that the file list may contain a mixture of files and groups of files.

Output Messages

The following messages will be output by the utility in the event of the corresponding error or condition.

Errors in Initial Parameters

The following messages are output, and the utility will terminate, if the initial parameters are incorrect or invalid.

1) ILLEGAL PARAMETER LIST

This message is displayed if the parameters supplied in the input message are incorrect. The utility will attempt to follow the message with a character string from the area of the input message which caused the failure.

2) INVALID CHARACTER IN IDENTIFIER < identifier >

This message is displayed if the < library-tape-name >, < disc-id >, or < file-id > entries in the input message contain characters which are illegal in identifiers.

3) < file-name > NOT FOUND

This message is displayed if the function was invoked through the LD * < file-name > format, and the parameter-file cannot be located.

4) DISK < disc-id > NOT AVAILABLE

This message is displayed if the specified disc is not on line.

5) NO SPECIFICATION GIVEN

This message is displayed if no parameters are entered following LD in the SCL input string.

Errors During Execution

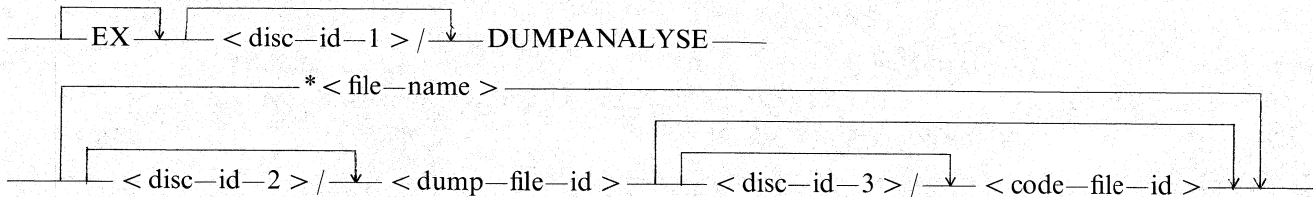
The following messages are displayed if the utility cannot perform some part of its function. The utility will continue to execute wherever possible.

- 1) **NO FILES IN THE FAMILY < group-id > ON DISK < disk-id > FOR DUMP**
This message is displayed if the utility is unable to find any files in the specified group on the disc. The utility will continue with the next file in the input parameter list.
- 2) **NO FILE < file-id > ON DISK < disk-id > FOR DUMP**
This message is displayed if a specified file is not present on the disc. The utility will continue with the next file in the input parameter list.
- 3) **< file-name > NOT DUMPED—IN OUTPUT USE**
This message will be displayed if a particular file is found to be in use. The tape will be purged and the utility will terminate.
- 4) **< file-name > NOT DUMPED—HAS BEEN REMOVED**
- 5) **< file-name > NOT DUMPED—HAS BEEN ALTERED**
This message is displayed if the contents of a file are altered between the start of a dump and the time that the file is to be copied to tape. The tape will be purged and the utility will terminate.
- 6) **< file-name > LOAD/DUMP DISCREPANCY**
This message will be displayed if end of file is reached before it is expected, and implies erroneous information in the Disk File Header.
- 7) **NO FILES TO DUMP**
This message is displayed if no files will be dumped to a tape. The utility will terminate.
- 8) **< file-name > DUMPED**
This message is displayed for each file dumped to tape.

Note that the above messages are output by the utility. Any applicable system message may be output by the MCP (for example **ERROR WHILE IN < verb >** if the tape cassette drive is opened while in use).

DUMPANALYSE

DUMPANALYSE (Analyse B80 Dump Files)



This utility provides the facility of analysing dump files produced by the B80 MCP through the SCL command "DP < mix-number > / < program-id >". Such a dump-file will have a file-id of DMFILEn where n is the mix number of the program before dumping.

Since the format of dump files is implementation dependent, this utility will only analyse B80 dump files. The utility may be executed from a disc other than the system disc by specifying < disc-id-1 >.

The utility will analyse dump files of programs which have been running using the interpreters "BILINTERP" or "COBOLINT". In order to produce a full analysis of a COBOL dump-file, both the dump-file and the code-file must be present. If the code-file is not present, no COP table data will be analysed. A full analysis of BIL/MPLII programs requires only the dump-file.

If the dump-file is on a disc other than the system disc, then < disc-id-2 > must be specified.

If the COBOL code-file is on a disc other than the system disc, then < disc-id-3 > must be specified. The < code-file-id > defaults to the program name held in the Program Parameter Block (PPB) of the dump-file, and is assumed to be on the same disc as the dump-file, if no code-file-name is specified.

On completion of the analysis, the dump-file will be closed with purge.

Analysis Output

When the utility is analysing a BIL or MPLII dump file, the following information is output to the line printer.

- 1) Information from the PPB of the program file which caused the dump. This information will include the program name, the interpreter name, the s-language, the compiler name, and compilation date, and also the reason for the dump.
- 2) An analysis of the files being used by the program file. This information will include the state of the file and, if the file is open, an analysis of the file information block including the file buffers.
- 3) An analysis of the BIL/MPLII S-registers.
- 4) A formatted analysis of the BIL/MPLII data and control stacks giving the dynamic procedure levels which had been entered at the time the dump was caused. This information will also include an analysis of descriptors and data of the identifiers of each entered procedure.

Note—a descriptor may be marked as illegal if the code which declares it has not yet been executed.

- 5) An unformatted analysis of all the accessible data segments used by the program.
- 6) An unedited analysis of the dump-file.

When the utility is analysing a COBOL or RPG dump file, the following information is output to the line printer.

- 1) Information from the PPB of the program file which caused the dump.
- 2) Partial analysis of the COBOL S-interpreter Work Area. This information will include the season for the dump, interpreter version number, last communicate response, and the current code segment pointer.

- 3) An analysis of the control stack.
- 4) An analysis of all the files being used by the program file. This information will include the state of the file, and, if the file is open, an analysis of the file information block including the file buffers.
- 5) An unformatted analysis of all the accessible data segments used by the program.
- 6) An analysis of the COBOL Current Operand (COP) table. This includes the COP number, data type, data length, segment number and digit displacement, whether it is indexed or subscripted, the index or subscript values, the table bound, and, if accessible, the contents of the COP.
- 7) An unedited analysis of the dump file.

Output Messages

The following messages will be output by the utility in the event of the appropriate error or inconsistency.

Warnings

- 1) ****WARNING** BIL DUMP FILE, SECOND PARAMETER IN INIT.MESS. IGNORED.**

This message will be displayed when the code—file—id has been specified for a BIL/MPLII dump—file.

Fatal Errors

The following messages will be output and the utility will terminate in the event of the appropriate error.

- 1) **PACK ID TOO LONG < disc—id >**
This message will be displayed when the < disc—id > exceeds seven characters.
- 2) **FILE ID TOO LONG < file—id >**
This message will be displayed when the < file—id > exceeds twelve characters.
- 3) **INVALID CHARACTERS IN FILE < file—name >**
This message will be displayed when invalid characters are encountered in the < disc—id > or the < file—id >.
- 4) **< file—name > NOT FOUND**
This message will be displayed when the < file—name > specified cannot be located.
- 5) **NO SPECIFICATION GIVEN**
This message will be displayed when no specifications are given in the initiating message.
- 6) **INTERPRETER VERSION NOT SUPPORTED < file—id >**
This message will be displayed when the interpreter name < file—id > from the PPB of the dump file is not recognised by the utility.

Note: the above messages are output by the utility. Any other applicable system message may be output by the MCP in addition to the above (for example, NO FILE if no printer is available).

EX

EX (Execute)

This intrinsic command initiates the loading and subsequent execution of the program identified by `< prog-id >`. The program is assumed to be on the system disc unless `< disc-id >` is specified. If the program is an MCS (Message Control System) then, in addition to loading the task, the data communication subsystem will be initialised. Only one MCS may exist in the mix, consequently, an attempt to execute a second program of that type will be rejected.

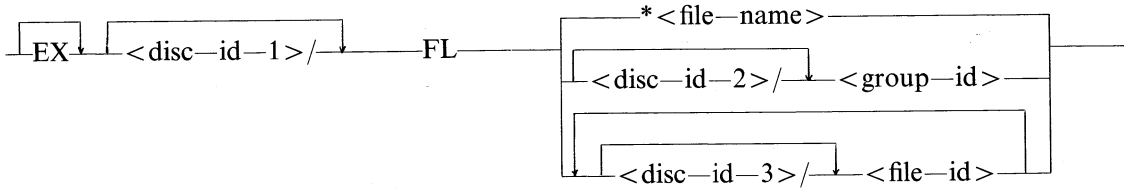
The `(< real-store >)` field is implementation dependent and indicates the magnitude in bytes of the immediate access storage (that is core memory, random access memory etc.) which is required for efficient execution of the program. This field is not required for the B80 implementation of CMS.

The `< text >` field is an optional character string which, if requested by the object program will be passed into the program's data space. If not requested, `< text >` will be discarded; it is not processed by the system.

Output Messages

One of the following messages will be displayed if the specified program cannot be loaded.

- 1) [50] LOAD FAILURE DISK NOT FOUND
The disc identified by `< disc-id >` cannot be found.
- 2) [51] LOAD FAILURE PROGRAM NOT FOUND
The program identified by `< prog-id >` cannot be found.
- 3) [52] LOAD FAILURE FULL MIX
The specified program's priority class is full, or the program currently executing requires that no other programs may co-exist.
- 4) [53] LOAD FAILURE NO USER DISK
There is insufficient available space on the specified disc for needed backing store.
- 5) [54] LOAD FAILURE INTERPRETER NOT FOUND
The interpreter file required to execute the program cannot be found.
- 6) [55] LOAD FAILURE USER COUNT ERROR.
The system only permits 7 concurrent users of a file: this includes program code files.
- 7) [56] LOAD FAILURE CODE FILE ERROR
The loader has detected an inconsistency in the code-file.
- 8) [57] LOAD FAILURE INVALID LOAD REQUEST
There is an error in the parameter of the SCL load request, for example, a thirteen character program-id.
- 9) [58] LOAD FAILURE INSUFFICIENT MEMORY
There is not enough main memory (core, RAM etc.) to hold this program's TCB and PCB.
- 10) [59] LOAD FAILURE MCS ALREADY PRESENT
Only one MCS may exist in the mix.
- 11) [60] LOAD FAILURE DUPLICATE PACK
The loader detected two or more discs on the system having the specified `< disc-id >`.
- 12) [61] LOAD FAILURE NULL MIX REQUIRED
The specified program must only be loaded if the system mix is empty and this is not the case.



This utility will display upon a Self Scan the properties of particular files or groups of files. The utility may be executed from a disc other than the system disc by specifying < disc-id-1 >. A group of files may be identified for analysis by specifying < group-id >. The group is assumed to be on the system disc unless < disc-id-2 > is specified. Alternatively, a list of files may be presented. Each file in the list is assumed to be on the system disc unless the respective < disc-id-3 > is specified.

If the input parameters specify a single file, or a group containing a single file then the properties of the file will be displayed, and the utility will terminate. If more than one file is to be analysed, then the utility will enable PK1 and PK6 after displaying the properties of the first file to be analysed. Depression of PK1 (or any OCK) will cause the attributes of the next file in the list or group to be displayed. The utility may be terminated by pressing PK6.

Output Format

The following information is displayed upon the Self Scan for each file which is found for analysis.

FL < disc-id-1 >/< file-id > < filetype > USING < integer-1 > OF < integer-2 > RECORDS
 RECSIZE < integer-3 >, BLOCKFACT < integer-4 > CREATED < julian-date >, ACCESSED
 < julian-date > AREA MAP: < 16 characters > OVERFLOW ON DISK < disc-id-4 >.

(OVERFLOW ON DISK will not be displayed if the file has no overflow areas allocated)

The < disc-id-1 > entry contains the name of the disc containing the file described by the display. The entry will contain the name of the system disc if no < disc-id-2 > or < disc-id-3 > was specified in the input parameters.

The < file-id > entry contains the name of the file described by the display. This entry will contain one of

- *RESERVED
- *AVAIL. TABLE
- *FILE DIREC
- *FILE HEADRS

identifying the components of the CMS disc directory, if the group-id identifies all files on a disc (that is, "FL < disc-id >/="), when these areas of the disc are analysed.

The < filetype > entry will contain one of

- SYSTEM (System file)
- CODE (Object code file)
- DATA (Normal Data file)
- SRCELANG (Source language file)

The actual file-size of the file described by the display is indicated by < integer-1 >; < integer-2 > identifies the maximum file-size specified for the file.

The number of characters per record is identified by < integer-3 >, and < integer-4 > indicates the number of records per block.

The creation date and last access date are shown in julian format (YYDDD) in the < julian-date > entries.

Sixteen characters are output at the end of the display to show the allocation of the 16 areas into which a file may be broken. Each character may be one of

- * Unallocated
- B allocated on this disc
- O allocated on overflow disc

The utility will display the message

FILE NAME = < disc—id > / < file—id > —NOT FOUND IN DISC DIRECTORY

on the Self Scan for any file which is not located.

The message

< disc—id > / < group—id > —NO FILES FOUND IN DIRECTORY FOR FAMILY

will be displayed upon the Self Scan if no files are found in the specified group.

Output Messages

The following messages will be output by the utility in the event of the corresponding error.

Errors in Initial Parameters

The following messages are displayed if there are errors in the initial parameters following FL in the SCL input message. The utility will terminate.

1) NO SPECIFICATION GIVEN

This message is displayed if no initial parameters are entered.

2) ILLEGAL PARAMETER LIST < string >

This message is displayed if the initial parameters are incorrect. The utility will attempt to follow the message with a number of characters from the area of the input message which caused the failure.

3) INVALID CHARACTER IN IDENTIFIER < string >

This message is displayed if the < disc—id >, < group—id >, or < file—id > entries in the input message contain characters which are illegal in identifiers.

4) < file—id > NOT FOUND

This message is displayed if the utility cannot find the parameter-file identified by the < file—name > macro-call.

Errors During Execution

1) DISK < disc—id > NOT OPENED—NOT ON LINE

This message is displayed if a disc identified by < disc—id—3 > in the input message cannot be found. The utility will continue scanning the parameters, ignoring all references to files on the identical disc.

Note that the above messages are output by the utility. Any applicable system message may be output by the MCP (for example NO FILE, if no Self Scan is available).



This utility removes all deleted records from a data file or from the data file portion of an indexed file with a consequent reconstruction of the key file portion of the indexed sequential file. Deleted records in the data contain the hexadecimal value FF in every byte.

The utility may be executed from a disc other than the system disc by specifying `<disc-id-1>`. The file to be squashed is identified by `<file-id>` and is assumed to be on the system disc unless `<disc-id-2>` is specified. If `<file-id>` refers to a key file, the `disc-id` and `file-id` of the associated data file are taken from parameters held in the key file, and the keyfile will be reconstructed so that it relates to the modified data file.

While the utility is running, no other task may gain access to the data file or the key file if one is specified.

Output Messages

The utility will output the following messages as appropriate during execution.

Normal End-of-Job

These messages are output after completion of all or part of the file squash.

- 1) `<file-name> SQUASHED FROM n RECORDS TO m RECORDS`
This message indicates the original and resulting filesizes of the data file. The integers `n` and `m` are decimal numbers of up to seven digits each.
- 2) `KEYFILE <file-name> RECONSTRUCTED`
This message is output in addition to the above if the file name specified to the utility identified a keyfile and the utility has successfully squashed the data file and reconstructed the key file.
- 3) `KEYFILE SORT FAILURE`
This message is output if SORT has indicated that it was not possible to properly construct a key file.

Errors in Initial Parameters

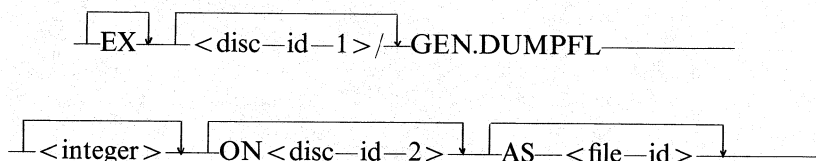
The following messages are displayed if there are errors or inconsistencies in the initial parameters. The utility will then terminate.

- 1) `<file-name> NOT ON LINE`
This message is displayed if the specified data file (or if a key file has been specified, the key file or the associated data file) cannot be located.
- 2) `ILLEGAL PARAMETER LIST`
This message is displayed if the initial parameters following "FS" in the SCL input string are incorrect. The utility will attempt to follow the message with a number of characters from the area of the input message which caused the failure.
- 3) `INVALID CHARACTER IN IDENTIFIER <identifier>`
This message is displayed if the `<disc-id>` or `<file-id>` portions of the file name contain characters which are illegal in identifiers.
- 4) `<file-name> NOT FOUND`
This message is displayed if the utility cannot find the parameter-file identified by the `* <file-name>` macro-call.
- 5) `NO SPECIFICATION GIVEN`
This message is displayed if no parameters are input following FS in the SCL input.

Note that the above messages are output by the utility. Any applicable system message may be output by the MCP (for example `ERROR WHILE IN <verb>`).

GEN.DUMPFL

GEN. DUMPFL (Create Empty Memory Dump File)



This utility will create an empty file on disc into which memory dumps can be made, and which can then be analysed by PMB80. The utility may be executed from a disc other than the system disc by specifying `< disc-id-1 >`.

If the optional `< integer >` is specified, it must be in the range 60 to 16383. The value refers to the filesize of the file to be created, and is in units of 1024 bytes (kilobytes). The default is 60.

If `< disc-id-2 >` is omitted, the system disc is assumed.

The file will be given the identifier "MEMDUMP" unless the optional `< file-id >` is specified.

Any previous file of the same identity will be removed.

Output Messages

The following messages may be displayed in the event of the corresponding condition or error.

Errors in Initial Parameters

The following messages will be displayed, and the utility will terminate, in the event of the corresponding error.

- 1) **ILLEGAL PARAMETER LIST** `< character-string >`
This message is displayed if the parameters following GEN. DUMPFL in the SCL input string are incorrect. The utility will attempt to identify the portion of the message containing the error through the `< character-string >`.
- 2) **SIZE TOO SMALL**
This message will be displayed if the `< integer >` specified is less than 60.
- 3) **SIZE TOO LARGE**
This message is displayed if the `< integer >` specified is larger than 16383.

Warnings

The following messages are displayed if the `< file-id >` or `< disc-id-2 >` entries are longer than permitted for the type of identifier. The entry will be truncated on the right.

- 1) **DISK NAME TOO LONG**
- 2) **FILE NAME TOO LONG**

Normal Termination

The utility will display the following message, and go to normal end of job, when the file has been created:

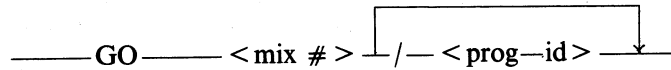
SPACE RESERVED

Note that the above messages are output by the utility. Any applicable system message may be output by the MCP (for example ERROR WHILE IN `< verb >`).

GO (Restart a previously stopped program)

GO

_____ GO _____ < mix # > / - < prog-id > _____



This function, implemented as an intrinsic within the MCP, allows the operator to restart a task which has either been stopped with the ST command, has issued a pause awaiting operator action, or is waiting for user disk space or special forms.

If the GO is accepted, the program will be allowed to continue executing. If the specified task is not waiting for a GO, the message

< mix # > / < prog-id > NOT STOPPED

is displayed and the GO is ignored.

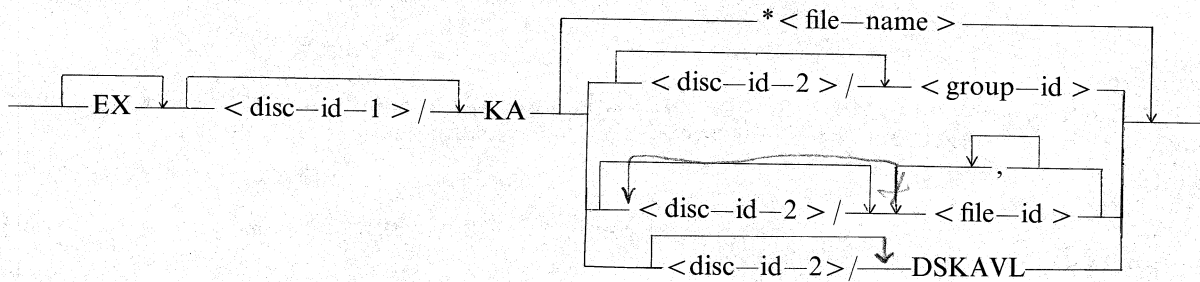
If the optional < prog-id > is specified in the command, then it must match with the mix number quoted otherwise the message

< mix # > / < prog-id > INVALID

is displayed.

KA

KA (Analyse Disc Space Assignment)



This utility provides a map of the usage of a particular disc in ascending disc address order in terms of areas and their assignment. The utility may be executed from a disc other than the system disc by specifying `< disc-id-1 >`. The disc which is to be analysed is identified by `< disc-id-2 >` or is the system disc if `< disc-id-2 >` is not specified. The utility will analyse the space assigned to a single file, a list of files, a group of files or will list the available space on a disc. Special reporting is given if the group identifies all files on the disc (that is `KA < disc-id > / =`). In addition to an analysis of the areas allocated to each file, this report will show the space assigned to the disc directory, any areas temporarily assigned (for example, a program's virtual memory space), available areas, and bad or missing areas. If files are created or deleted by the system during an execution of KA the map will be inaccurate. It is therefore recommended that the utility be executed when no other programs are in the mix. The analysed output will be to a line printer (or a console printer if no line printer is available) and will list the areas in ascending disc address order, associating with each area its initial sector address, its length in sectors, and its status which will be allocated, available, temporary, or bad. If the area is allocated, the file-id of the file to which the area is assigned will also be listed. If the reserved file-id "DSKAVL" is specified alone, then an analysis of the available space on the disc identified by `< disc-id-2 >` will be printed. Use of the file-id "DSKAVL" in a list of files will cause the utility to search the directory for a file of that name, in the normal manner.

Output Format

Four columns of information will be output to the printer. The column headings, the format of the values these columns contain, and the significance of these values is as follows:—

<u>HEADING</u>	<u>VALUE</u>	<u>SIGNIFICANCE</u>
AREA ADDRESS	8 digits @ 6 digits @	Sector Address of start of Area
AREA LENGTH	8 digits @ 6 digits @	Number of Sectors in this Area
STATUS	9 characters	See Note 1
FILE NAME	12 characters	See Note 2

(The area address and area length will also be printed in hexadecimal format alongside the decimal values).

Note 1 The status will be one of AVAILABLE, ASSIGNED, TEMPORARY, BAD, or *MISSING* depending on whether the area is available, allocated to a file, denoted as temporary, unusable, or lost.

Note 2 If the area status is ASSIGNED then this field will contain the identifier of the file which has the area allocated to it.

The status *MISSING* occurs if an area is not referenced from anywhere within the file directory or available table. This may be because the area is in fact lost, or because existing files have been opened, have had further areas allocated to them and are still open during the execution of the utility.

If an area is contained partly or completely in an area previously listed, the message:—

AREA APPARENTLY ASSIGNED TWICE

will be printed with the area information.

If an area is assigned beyond the addressable space then the message

AREA ASSIGNED BEYOND MAXIMUM ADDRESS

will be printed with the area information.

If the utility finds that files are open on the disc then the message:—

NOTE: OUTPUT FILES ON DISK WERE OPEN DURING THIS EXECUTION OF KA

will be printed at the end of the KA listing.

If a specified < file—id > is not found then the message

—NOT FOUND IN DIRECTORY ON THIS DISC

is printed to the right of the file—name column in the KA listing.

If a < group—id > is specified, and no files in that group are found, then the message

—NO FILES IN DIRECTORY FOR THIS FAMILY

is printed to the right of the file—name column in the KA listing.

Output Messages

The following messages may be output by the utility in the event of the appropriate error. The utility will then terminate.

1) **ILLEGAL PARAMETER LIST**

The character string following “KA” in the SCL input message is incorrect. The utility will attempt to follow the message with a character string from the area of the message which caused the failure.

2) **INVALID CHARACTER IN IDENTIFIER < identifier >**

This message is printed if a character which is not allowed in identifiers is found in the < disc—id > or < file—id > of the specified file.

3) **DISK < disc—id > NOT OPENED—NOT ON LINE**

This message is displayed if the disc identified by < disc—id > has not been installed on the system.

4) **TABLE SIZE EXCEEDED**

This message is displayed if the number of lines of output required is greater than that permitted by the utility.

5) **< file—name > NOT FOUND**

This message is displayed if the utility cannot find the parameter—file identified by the * < file—name > macro-call.

6) **NO OUTPUT GENERATED BY KA**

This message is displayed if there is no output for printing.

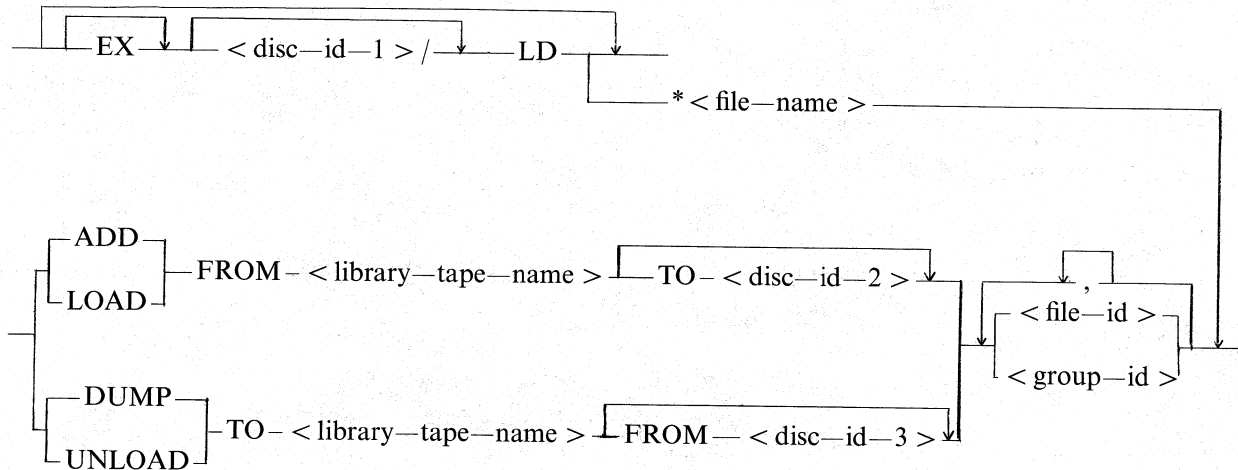
7) **NO SPECIFICATION GIVEN**

This message is output if no parameters are entered following KA in the SCL input string.

Note that the above messages are output by the utility. Any applicable system message may be output by the MCP (for example NO FILE if no printer is available).

LD

LD (Tape Library Utility)



This utility provides library tape maintenance facilities through the four functions ADD, LOAD, DUMP, and UNLOAD. LD will automatically be invoked if these mnemonics are used without the preceding LD, therefore the `<file-id>` of LD should not be changed if this facility is required. LD must be initiated by name if the `* <file-name>` construct is to be used, or if the utility is to be executed from a disc other than the system disc.

LOAD and ADD provide the capability of copying files or groups of files from a library tape to the disc identified by `<disc-id-2>`, or the system disc if `<disc-id-2>` is omitted. The `<library-tape-name>` following "FROM" identifies the library tape containing the file or group of files to be copied to the disc. The file-list identifies the particular files or groups of files to be copied from the library tape. Note that the file list may contain a mixture of files and groups of files.

If LOAD is specified, then the files are copied to the disc and any duplicates are removed. If ADD is specified, then only those files which do not have copies already on the disc are loaded.

DUMP and UNLOAD provide the capability of copying files or groups of files from the disc specified by `<disc-id-3>` (or the system disc if `<disc-id-3>` is omitted) to a library tape. The `<library-tape-name>` following "TO" specifies the name which will be given to the library tape. The file list identifies the particular files or groups of files to be copied to the tape. Note that the file list may contain a mixture of files and groups of files.

If UNLOAD is specified, then the files are deleted from the disc after they have been copied to the tape. If DUMP is specified, the disc is not altered.

Output Messages

The following messages may be output by the utility in the event of the corresponding error or condition.

Errors in Initial Parameters

The following messages will be output, and the utility will terminate, if the initial parameters following LD in the SCL input string are invalid.

1) ILLEGAL PARAMETER LIST

This message will be displayed if the input parameters are incorrect. The utility will attempt to follow the message with a number of characters from the area of the input message which caused the failure.

2) INVALID CHARACTER IN IDENTIFIER `<identifier>`

This message is displayed if the `<library-tape-name>`, `<file-id>`, or `<disc-id>` entries in the input message contain characters which are illegal in identifiers.

3) < file—name > NOT FOUND

This message is displayed if the utility cannot locate the parameter file identified by the * < file—name > construct.

4) DISK < disc—id > NOT AVAILABLE

This message is displayed if the specified disc is not on line.

5) NO SPECIFICATION GIVEN

This message is displayed if no parameters are entered following LD in the SCL input string.

Normal Execution

The following messages are displayed as each file is successfully loaded, unloaded, or added.

1) < file—name > REMOVED

This message is displayed for each file removed as a result of LOAD or UNLOAD.

2) < file—name > LOADED

This message is displayed for each file loaded or added.

3) < file—name > DUMPED

This message is displayed for each file dumped or unloaded.

Errors During Execution

The following messages are displayed if the utility cannot perform some part of its function. The utility will continue to execute wherever possible.

1) < library—tape—name > NOT A RECOGNISED DUMP TAPE

This message is displayed if the tape which has been provided for loading does not have a recognisable header. The correct tape should be provided or the utility should be DS'ed (discontinued). If the tape is provided, the GO command may be required in some CMS implementations in order to restart the utility.

2) NO FILES IN THE FAMILY < group—id > ON { TAPE
DISK } < identifier > FOR { ADD
LOAD
DUMP
UNLOAD }

This message is displayed if the utility is unable to find any files in the specified group on the specified disc or tape. The utility will continue with the next file in the input parameter list.

3) NO FILE < file—id > ON { TAPE
DISK } < identifier > FOR { ADD
LOAD
DUMP
UNLOAD }

This message is displayed if a particular file is not present on the library tape or on the disc. The utility will continue with the next file in the input parameter list.

4) < file—name > NOT DUMPED—IN OUTPUT USE

This message will be displayed for each file found to be open by an unload or dump. The file will not be dumped. The tape will be purged and the utility will terminate.

5) < file—name > NOT DUMPED—HAS BEEN REMOVED

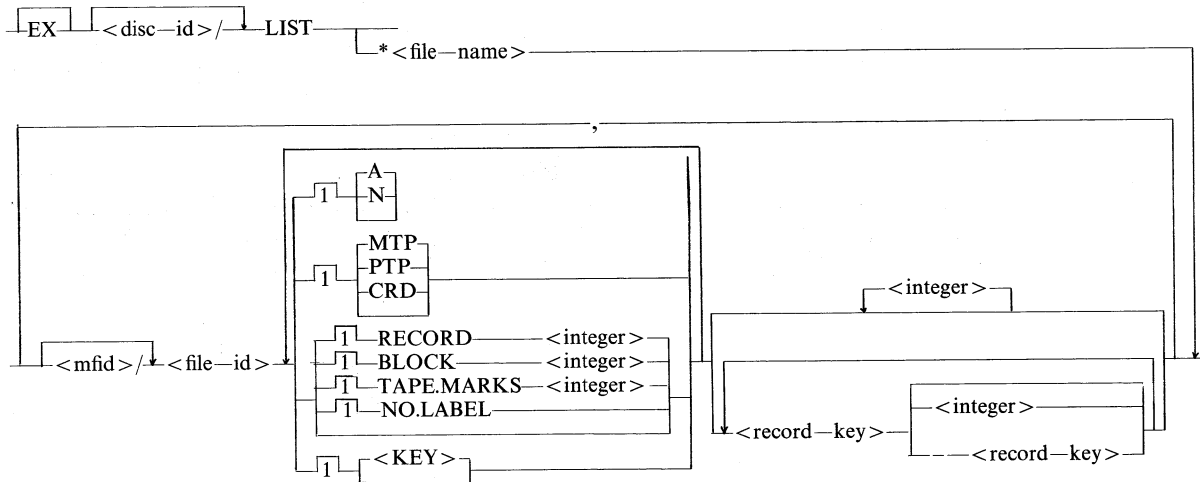
This message will be displayed if a file has been removed between the start of a dump or unload and the time that the file is to be copied to tape. The tape will be purged and the utility will terminate.

6) < file—name > NOT DUMPED—HAS BEEN ALTERED

This message is displayed if the contents of a file are changed between the start of a dump or unload and the time that the file is copied to tape. The tape will be purged and the utility will terminate.

- 7) < file—name > LOAD/DUMP DISCREPANCY
This message will be displayed if end of file is reached before it is expected, and implies erroneous information in the Disk File Header.
- 8) NO FILES TO LOAD
This message is displayed if no files will be dumped to a tape. The utility will terminate.
- 9) NO FILES TO DUMP
This message is displayed if no files will be dumped to a tape. The utility will terminate.
- 10) < file—id > NOT LOADED—ALREADY ON DISK
This message is displayed during an ADD function for each requested file on the tape which is not loaded because a file with the same identity already exists on disc.
- 11) ALTHOUGH WITH DIFFERENT ATTRIBUTES
This message is displayed immediately following the ALREADY ON DISC message (above) if the two files differ in record size, block size, file size, or file type. The message is for information only.

Note that the above messages are output by the utility. Any applicable system message may be output by the MCP (for example NO FILE if no purged output tape is available).



This utility gives extensive file listing capabilities. It allows a file or a series of files on any CMS device or devices to be displayed in whole or in part, and in various formats, on a line printer (or console printer if no line printer is available). The utility may be executed from a disc other than the system disc by specifying `< disc-id >`.

A string of LIST specifications may be given, each specification separated by a comma. If the utility is unable to perform a particular request in the specification list, it will proceed to the next request in the list. Each field in a list request is described below.

File Name

The `< mfid >` applies only to disc or magnetic tape (including cassette) files. For disc files, if `< mfid >` is not specified, the system disc is assumed. For magnetic tape and cassette files, if `< mfid >` is not specified, a single file tape is assumed. The `< file-id >` identifies a particular file to be listed.

Print Format

The output format of the listing may be specified as alpha (A), numeric (N) or alphanumeric (no entry). A ten character column on the left hand side of the printer page is reserved for the record number. This field is left empty when a file is listed in key order via an associated key-file (that is, an indexed file, see device type DISK). For indexed files, the record key is printed on a separate line using the requested print format (A, N, both) prior to the printing of each data record. The data records will be printed to the right of the record number field (which may be empty) as follows. If A is specified each record will be output in 100 character lines across the page until the complete record is printed. If N is specified, then each byte of the record is printed as two hexadecimal digits, 80 characters to the line, until the complete record is printed. If neither A nor N is specified, then each record is output in both character and hexadecimal formats, 32 bytes per line, until the complete record is printed. The first 32 print positions will display the character representation of each byte. The next four print positions are spaces. The next 64 print positions are used to display the hexadecimal equivalent of the 32 characters printed at the start of the line.

The following points should be noted with regard to each CMS device type.

Paper Tape

Any labelled CMS paper tape file may be listed by specifying PTP. The format option (A, N) determines the output mode. The record size is assumed to be 180 characters. Other options (for example, RECORD `< integer >`, or `< KEY >`) are invalid for paper tape files.

Punched Card

Any labelled CMS punched card file may be listed by specifying CRD. The format option (A, N) determines the output mode. The record size is assumed to be the same as the physical record size of the device (that is, 80 or 96 characters). Other options (for example, RECORD < integer >, or < KEY >) are invalid for punched card files.

Magnetic Tape and Cassette

Any magnetic tape or cassette file may be listed by specifying MTP. If the file is not a labelled CMS file (either unlabelled or containing non-CMS labels) then it may be listed by using the unlabelled option described below. The format option (A, N) determines the output mode. The RECORD, BLOCK, TAPE. MARKS, and NO. LABEL options may be used when listing magnetic tape and cassette files. The < KEY > option does not apply to magnetic tape and cassette.

Record and Block

RECORD and BLOCK may be used to specify the number of *characters* in each record and block in the file. The record and block sizes are determined as follows.

LABELLED CMS FILES.

RECORD and BLOCK need not be specified when the record size is less than 1024 characters, since the utility will take the values from the file label. RECORD and BLOCK must be specified when the record size is greater than 1024 characters, otherwise the file will not be listed. The record and block sizes used are chosen in the following order

RECORD < integer > *then* label value of record size.

BLOCK < integer > *then* RECORD < integer > *then* label value of block size.

UNLABELLED FILES.

The utility defaults to block and record sizes of 180 characters unless the RECORD and BLOCK options are specified. The record and block sizes used are chosen in the following order

RECORD < integer > *then* 180 characters.

BLOCK < integer > *then* RECORD < integer > *then* 180 characters.

Note

Care should be taken to ensure that the record and block sizes specified are compatible with the physical block size on the tape. The block size selected must be an integer multiple of the record size, therefore the defaults must be considered when specifying *only* RECORD *or* BLOCK. The utility will attempt to identify inconsistencies when using labelled CMS files. Any inconsistency not isolated by LIST will cause MCP to discontinue (DS/DP) the utility.

No. Label

The NO. LABEL option is provided to permit the LISTing of unlabelled (or labelled non-CMS) files. The < file—id > entry must be included in order to identify the file for SCL purposes.

MCP will output the message

“< mx >/LIST < 14 > WAITING UNLAB SPURIUS/< file—id > MT DEVICE REQUIRED”

An “AD < mx >/LIST < tape—peripheral >” message must be used to identify the labelled file. The end of file recognition for unlabelled files is determined by tape mark count as described below.

Tape.Marks

The TAPE. MARKS option allows the user to specify the aggregate numbers of tape marks which will indicate end of file to the utility when listing an unlabelled file. The default value is 2. Each tape mark which is encountered will contribute to this total, thus a standard labelled CMS file will be listed up to, but excluding, the trailing label if NO. LABEL and two tape marks are specified. (A labelled CMS file consists of “Label TM data TM label”). The user must therefore be aware of the format of any file which is to be listed when using the NO. LABEL option.

Disk

Any CMS disc file may be listed. The disc device type is specified by omitting the device specifier. The format option (A, N) determines the output mode. Record and block sizes are taken from the disc file header. The RECORD, BLOCK, TAPE, MARKS, and NO. LABEL options are ignored if specified. Indexed files may have their keyfile or data file portions listed by use of the < KEY > option described below.

Indexed Files

The utility assumes that a listing of the associated data file is required (in key order) when the file identified by < file—id > is recognised as a keyfile. If a listing of the contents of a keyfile is required, then the < KEY > option must be specified.

When listing a data file via a keyfile, the < record—key > options of the selective list specification may be used as described below.

Partial File List

Listings of selected portions of a file may be requested via the selective list specification. The various formats of selective list specification apply to disc and non-disc files as follows.

Non-disc Files

The < integer—list > may be specified for paper tape (PTP), punched card (CRD), magnetic tape and cassette (MTP). Each < integer > is a decimal number of up to 7 digits, and each pair of < integer > s in the string specify a starting record number and the number of records required from that point respectively. Relative record numbers commence with record 1. If there is an odd number of < integer > s, then the last < integer > is equivalent to a pair of < integers > which would cause LIST to process to the end of the file. The < integer—list > must not specify any random access of the file, that is, the first < integer > of each < integer > pair must identify a higher record number than any previously listed record.

The < record—key > option is not applicable to non-disc files.

Disc Files

The < integer > list applies to all disc files and has identical use to that described above for non-disc devices, except that random access is permitted when using disc.

The < record—key > option allows selective listing of an indexed file by reference to the key value contained in each record. The user may specify a number of records to be listed via the < integer > option, or a terminating key value via the < record—key > option. The listing will include the records containing the starting and terminating values of < record—key > .

Output Messages

The following messages may be output by the utility in the event of the corresponding error or condition.

Errors in Initial Parameters

The following messages will be displayed, and the utility will terminate, if there are errors in the initial parameters following LIST in the SCL input message.

1) **ILLEGAL PARAMETER LIST**

This message is displayed if the initial parameters are incorrect or invalid. The utility will attempt to follow the message with a number of characters from the area of the input message which caused the failure.

2) **INVALID CHARACTER IN IDENTIFIER < identifier >**

This message is displayed if the < mfid > or < file-id > entries in a request contain characters which are illegal in identifiers.

3) **NO SPECIFICATION GIVEN**

This message is displayed if no parameters are entered following LIST in the SCL input string.

Errors During Execution

The following messages are displayed if the utility is unable to perform some part of its function. The utility will continue to execute wherever possible.

1) **< file-name > NOT FOUND**

This message is displayed if the specified file cannot be located. The utility will proceed to the next request in the specification string, unless the missing file was identified as containing the input parameters (* file-id), when the utility will go to end-of-job.

2) **< integer >, < integer > IN < file-name > NOT LISTED**

This message is displayed if a record number in an < integer > pair indicates a section of the file at a lower file address than a previously specified section. The < integer > pair is ignored.

3) **< file-name > EXHAUSTED DURING < integer >, < integer >**

This message is displayed if end of file is encountered while the section of the file indicated by the < integer > pair is being listed. The utility will proceed to the next request in the specification string.

4) **< file-name > EXHAUSTED DURING RANGE < key-1 > { - < key-2 > } < integer >**

This message is displayed if end of file is encountered while the section of the file indicated by < key-1 > < integer > is being listed, or if no records were found in the range < key-1 > - < key-2 >.

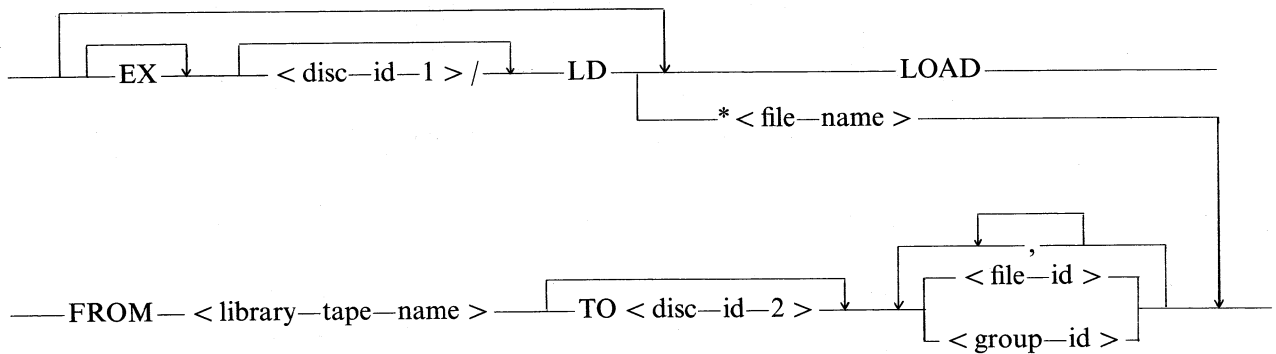
5) **NO RECORDS FOR LISTING FROM < file-id >**

This message is displayed if the identified file contains no records for listing.

6) **< file-id > NOT ACCEPTABLE—RECORD SIZE OF < integer > IS GREATER THAN THE MAXIMUM SPECIFIED FOR THIS RUN—RESUBMIT**

This message will be displayed if a file is submitted to the utility with a record size greater than that expected. This can happen if an MTP file with a record size greater than 1024 characters is submitted to the utility without the record size being properly specified in the parameters. This request should be re-specified, since this invocation of the utility will ignore it and continue with the next request if any.

Note that the above messages are output by the utility. Any applicable system message may be output by the MCP (for example, NO FILE if no printer is available).



Note that this function is a sub-program within the utility LD. MCP recognises the mnemonic LOAD if LD is not specified, and will automatically initiate a load of the LD utility. The <file-id> of LD should not therefore be changed if the LOAD function is to be invoked in this manner. To discontinue the LOAD function, "DS <mix-number> / LD" must be used.

This function may be executed from a disc other than the system disc by specifying LD and <disc-id-1>.

The function provides the capability of copying files or groups of files from a "library tape" to the disc specified by <disc-id-2> (or the system disc if <disc-id-2> is omitted). A library tape is defined to be a tape output by the DUMP or UNLOAD functions. The <library-tape-name> identifies the library tape containing the files or groups of files to be copied to the disc. The file list identifies the particular files or groups of files to be copied from the library tape. Note that the file list may contain a mixture of files and groups of files. Any corresponding files which are already on the disc will be removed.

Output Messages

The following messages will be output by the utility in the event of the corresponding error or condition.

Errors in Initial Parameters

The following messages are output, and the utility will terminate, if the initial parameters are incorrect or invalid.

1) ILLEGAL PARAMETER LIST

This message is displayed if the parameters supplied in the input message are incorrect. The utility will attempt to follow the message with a character string from the area of the input message which contains the errors.

2) INVALID CHARACTER IN IDENTIFIER <identifier>

This message is displayed if the <library-tape-name>, <disc-id>, or <file-id> entries in the input message contain characters which are illegal in identifiers.

3) <file-name> NOT FOUND

This message is displayed if the function was invoked through the LD * <file-name> format, and the parameter-file cannot be located.

4) DISK <disc-id> NOT AVAILABLE

This message is displayed if the specified disc is not on line.

5) NO SPECIFICATION GIVEN

This message is displayed if no parameters are entered after LD in the SCL input string.

Normal Execution

The following messages are displayed for each file which is successfully loaded

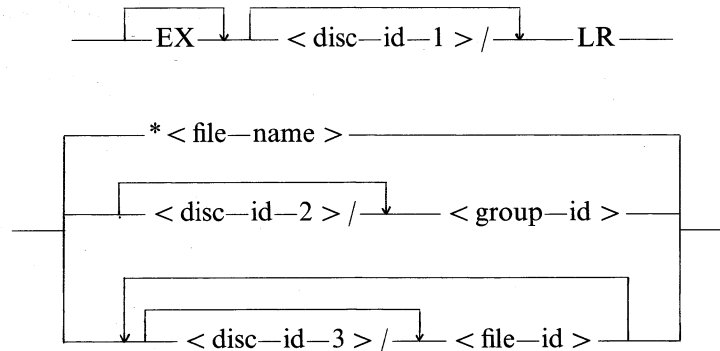
- 1) < file—name > LOADED
- 2) < file—name > REMOVED
This message is displayed if there was a corresponding file already resident on the disc. The old version is removed.

Errors During Execution

The following messages are displayed if the utility cannot perform some part of its function. The utility will continue to operate wherever possible.

- 1) < library—tape—name > NOT A RECOGNISED DUMP TAPE
This message is displayed if the tape which has been provided for loading does not have a recognisable header. The correct tape should be provided or the utility should be DS'ed (discontinued). If the tape is provided, the GO command may be required in some CMS implementations in order to restart the utility.
- 2) NO FILES IN THE FAMILY < group—id > ON TAPE < library—tape—name > FOR LOAD
This message is displayed if the utility is unable to find any files in the specified group on the library tape. The utility will continue with the next file in the input parameter list.
- 3) NO FILE < file—name > ON TAPE < library—tape—name > FOR LOAD
This message is displayed if a specified file is not present on the library tape. The utility will continue with the next file in the input parameter list.
- 4) < file—name > LOAD/DUMP DISCREPANCY
This message will be displayed if end of file has been reached before it is expected, and implies erroneous information in the Disk File Header. The utility will continue with the next file in the input parameter list.
- 5) NO FILES TO LOAD
This message is displayed if no files will be loaded from the tape. The utility will terminate.

Note that the above messages are output by the utility. Any applicable system message may be output by the MCP (for example ERROR WHILE IN < verb > if the tape cassette drive is opened while in use).



This utility provides for the generation of a listing on a printer of the properties of particular files or a group of files. The utility may be executed from a disc other than the system disc by specifying `< disc-id-1 >`. A group of files may be identified for analysis by specifying `< group-id >`. The group is assumed to be on the system disc unless `< disc-id-2 >` is specified. Alternatively, a list of files may be presented. Each file in the list is assumed to be on the system disc unless the respective `< disc-id-3 >` is specified.

For each file identified in the input specification, the following attributes are analysed:

- Number of Records in the file
- Record Size
- Block Size
- Creation Date
- Last Access Date
- File Type
- Area Sizes

If a particular file is not located, this will be indicated on the output listing.

Output Format

Twelve columns of information will be output to the printer for each disc for which information is requested. The column headings, the format of the values which these columns contain, and the significance of these values is as follows:

<u>HEADING</u>	<u>VALUE</u>	<u>SIGNIFICANCE</u>
FILE NAME	12 characters	File identifier
ACTUAL SIZE	7 digits	The number of records in the file
MAXIMUM SIZE	7 digits	The maximum number of records which may be written to the file
RECORD SIZE	5 digits	The number of characters in the record
RECORDS PER BLOCK	5 digits	The number of records in each block
CREATION DATE	5 digits	File Creation Date (Julian, YYDDD)
LAST ACCESS DATE	5 digits	Last Access Date (Julian, YYDDD)
FILE TYPE	8 characters	See Note 1
NO. AREAS	2 digits	Number of Areas currently allocated
OVERFLOW DISC	7 characters	See Note 2
AREA ADDRESSES	8 digits @ 4 digits @	Starting sector address of each allocated area in decimal and hexadecimal. See Note 3
AREA SIZES	8 digits @ 4 digits @	The length on sectors of each allocated. See Note 3

Note 1 The FILE TYPE entry will be one of:—

DATA — Normal Data File
CODE — S-CODE File
KEY — Key File (see Note 4 below)
SYSTEM — System File (interpreter, etc.)
SRCELANG — Source Language File
SRCELIBR — Source Library File

Note 2 If a file has areas allocated on an overflow pack the disc—id of the overflow pack will be listed.

Note 3 For areas on overflow packs, the characters “OVF” will follow the size.

Note 4 If a file is found to be a Key File, then a second line will be printed giving information about the Data File with which it is associated.

Output Messages

The following messages will be output by the utility in the event of the corresponding error.

Errors in Initial Parameters

The following messages are displayed if there are errors in the initial parameters following LR in the SCL input message. The utility will terminate.

1) **ILLEGAL PARAMETER LIST**

This message is displayed if the initial parameters are incorrect. The utility will attempt to follow the message with a number of characters from the area of the input message which caused the failure.

2) **INVALID CHARACTER IN IDENTIFIER < identifier >**

This message is displayed if the < disc—id >, < group—id >, or < file—id > entries in the input message contain characters which are illegal in identifiers.

3) **< file—name > NOT FOUND**

This message is displayed if the utility cannot find the parameter—file identified by the * < file—name > macro-call.

4) **NO SPECIFICATION GIVEN**

This message is displayed if no parameters are entered after LR in the SCL input string.

Errors During Execution

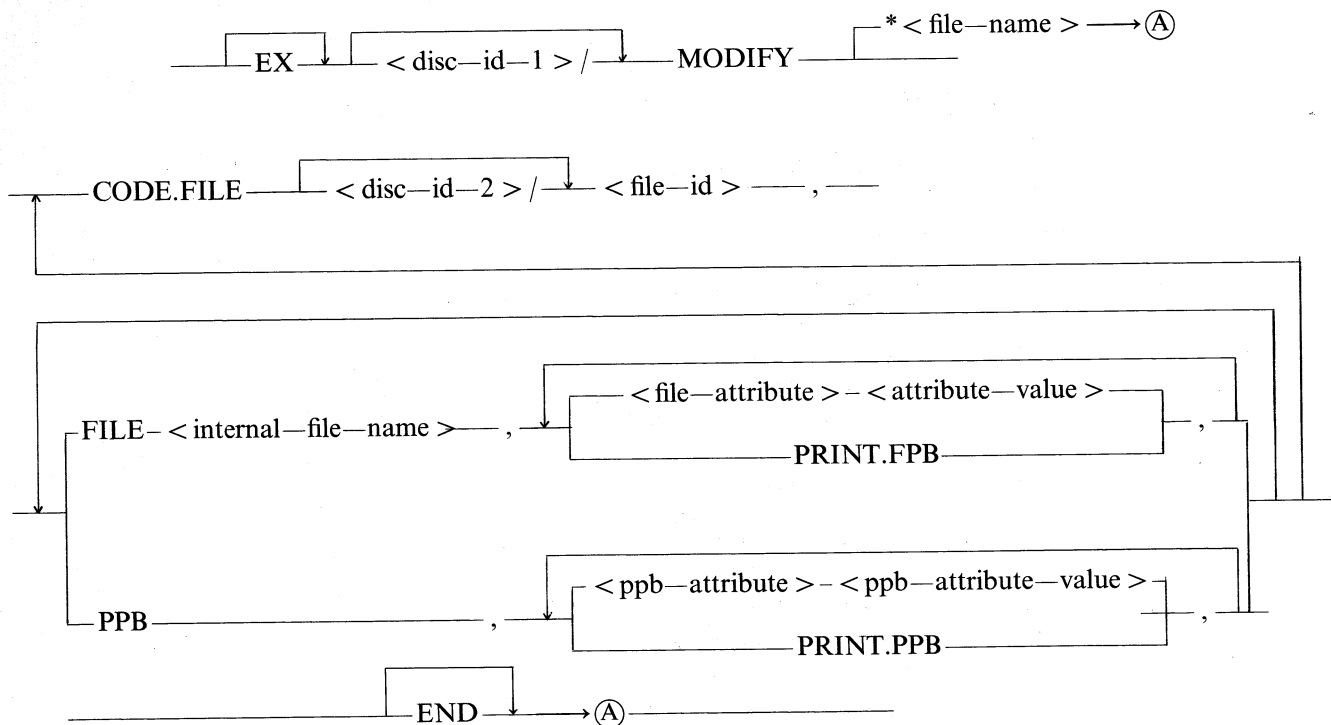
1) **DISK < disc—id > NOT OPENED—NOT ON LINE**

This message is displayed if a disc identified by < disc—id—3 > in the input string cannot be found. The utility will continue scanning the parameters, ignoring all references to files on the identified disc.

Note that the above messages are output by the utility. Any applicable system message may be output by the MCP (for example NO FILE if no printer is available).

MODIFY (Program File Modification)

MODIFY



Note: this utility should not be used until the information presented in the CMS Master Control Program (MCP) Reference Manual, form number 2007555, is thoroughly understood. The utility is primarily intended for Burroughs Field Support personnel, and is included for the benefit of users who are prepared to acquire the necessary depth of knowledge.

This utility allows a number of attributes within the file parameter block (FPB) and program parameter block (PPB) of program files on disc to be modified. The utility may be executed from a disc other than the system disc by specifying <disc-id-1>.

Each program codefile to be modified is identified by <file-id>, and is assumed to be on the system disc unless <disc-id-2> is specified. Any number of code files may be modified during one execution of MODIFY.

If modifications are to be made to an FPB within a program code file, then the particular FPB is identified by specifying <internal-file-name>. Each <internal-file-name> of a CMS program may be up to 30 characters in length, and can be found from a source listing of the appropriate program. Any number of FPB's may be modified in each code file.

A list of attributes and their new values may be presented in any order within a FILE clause, and the utility may be directed to produce an analysed printer listing of the appropriate FPB by specifying PRINT.FPB. Note that the modification string is processed sequentially, so the printer listing will reflect the state of the FPB after any modifications preceding the print request, but before any modifications succeeding the print request. The following attributes may be modified by specifying the identifier indicated under <file-attribute>, followed by a legal value as shown under <file-attribute-value> :-

<file-attribute>	<file-attribute-value>
MFID	Up to 7 alphanumeric characters
FID	Up to 12 alphanumeric characters
REEL	3 decimal digits less than or equal to 999

< file—attribute >

< file—attribute—value >

DEVICE	PR (any Printer)
	KP (Keyboard Printer)
	KD (Keyboard Display)
	KB (Keyboard any output)
	SP (Serial Printer)
	LP (Line Printer)
	CR (Any Card Reader)
	CP (Any Card Punch)
	CRP (Any Card Reader-Punch)
	CR80 (80 Column Card Reader)
	CP80 (80 Column Card Punch)
	CRP80 (80 Column Card Reader-Punch)
	CR96 (96 Column Card Reader)
	CP96 (96 Column Card Punch)
	CRP96 (96 Column Card Reader-Punch)
	PTR (Paper Tape Reader)
	PTP (Paper Tape Punch)
	MT (Magnetic Tape Reel or Cassette)
	MT9 (Magnetic Tape Reel)
	CS (Magnetic Tape Cassette)
	MT9IN (Magnetic Tape Reel without Write Permit)
	(Magnetic Tape Cassette without Write Permit)
	DC (any Disc)
RECORD	Up to 5 decimal digits less than or equal to 65535 ($2^{16}-1$)
BUFFER	Up to 5 decimal digits less than or equal to 65535 ($2^{16}-1$)
FILESIZE	Up to 7 decimal digits less than or equal to 1048560 ($2^{20}-16$)
NO.BUFFERS	Up to 2 decimal digits less than or equal to 16
CYCLE	Two decimal digits
FORMS	ON or OFF
SET.UPDATE	ON or OFF
NO.LABEL	ON or OFF
CONDITIONAL	ON or OFF
SINGLEAREA	ON or OFF
GEN.CHECK	ON or OFF
NO.REWIND	ON or OFF
CLOSEMODE	LOCK, PURGE, REMOVE, RELEASE, or HALF.CLOSE
CRUNCH	ON or OFF
MERGE	ON or OFF
OTHERUSE	FREE, LOCK.ACCESS, or LOCKED
MYUSE	INPUT, OUTPUT, or IO
EXTEND	ON or OFF
ACCESSMODE	SEQUENTIAL, STREAM, or RANDOM
GEN.NO	Up to 5 decimal digits less than or equal to 65535 ($2^{16}-1$)
LAST.ACCESS	Five decimal digits
SAVE	Up to 3 decimal digits less than or equal to 999
FILE.DEFAULT	TYPE 1 thru TYPE 29. Refer to the CMS MPLII Reference Manual form number 2007563
D.MFID	Up to 7 alphanumeric characters
D.FID	Up to 12 alphanumeric characters
ROUGH.TABLE	Up to 5 decimal digits less than or equal to 65535 ($2^{16}-1$)
KEY.LENGTH	Up to 3 decimal digits less than or equal to 255 (2^8-1)
KEY.OFFSET	Up to 5 decimal digits less than or equal to 65535 ($2^{16}-1$)

The last 5 attributes above are only applicable to Indexed Files.

Refer to the CMS Master Control Program (MCP) Reference Manual, form number 2007555 for details of the significance of the above attributes.

The PPB clause allows limited modification of the execution environment specified by the program parameter block. A list of the permitted attributes may be presented in any order within a PPB clause, and the utility may be directed to produce an analysed printer listing of the PPB by specifying PRINT.PPB.

Note that the modification string is processed sequentially, so the printed listing will reflect the state of the PPB after modifications preceding the print request but before any modifications succeeding the print request. The following attributes may be modified by specifying the identifier indicated under < ppb—attribute >, followed by a legal value as shown under < ppb—attribute—value > :—

< ppb—attribute >	< ppb—attribute—value >
INTERP.PACK	Up to 7 alphanumeric characters
INTERP.NAME	Up to 12 alphanumeric characters
CLASS	A or B or C
EOJ.SUPPRESS	ON or OFF

Refer to the CMS Master Control Program (MCP) Reference Manual, form number 2007555 for details of the significance of the above attributes.

Output Messages

The following messages may be displayed by the utility if the input parameters are incorrect. The utility will terminate.

- 1) **ILLEGAL PARAMETER LIST** < character string >
This message is displayed if file—id or disc—id containing characters illegal in identifiers is found. inconsistent. The utility will attempt to identify the area containing the errors by presenting a < character string > from the message.
- 2) **INVALID CHARACTER IN IDENTIFIER** < identifier >
This message is displayed if file—id or disc—id containing characters illegal in identifiers is found.
- 3) **NO SPECIFICATION GIVEN**
This message is displayed if no parameters are entered following MODIFY in the SCL input string.
- 4) < file—id > **NOT FOUND**
This message is displayed if the parameter file to be used for the input message cannot be located.

The following messages may be displayed during the execution of the utility.

- 1) **ATTRIBUTE VALUE MISSING**
The attribute value is either missing or incorrect.
- 2) **KEYWORD IN ERROR**
Either the attribute name or the value name is not correct.
- 3) **ATTRIBUTE—VAL INCONSISTENT**
The attribute being assigned to cannot accept the value being given.
- 4) **INCORRECT ATTRIBUTE**
A value is being assigned to a value, rather than to an attribute.

- 5) **DEVICE—MYUSE INCONSISTENT**
The value of MYUSE and the value of DEVICE are incompatible, and will cause an error.
- 6) **FILE—SIZE TOO LARGE**
The file size is greater than 2²⁰–16.
- 7) **TOO MANY BUFFERS**
The value assigned to NO.BUFFERS is greater than 16.
- 8) **REC. NOT INTEGRAL OF BUF.**
The buffer size is not an integer multiple of the record size.
- 9) **CODE FILE NAME IN ERROR**
The format of the code—file—name is incorrect. All modifications to this codefile are ignored.
- 10) **FILE NAME NOT FOUND**
The name of the file associated with the FILE clause does not exist in the code file being modified. All modifications to this file are ignored.
- 11) **MISSING SEPARATOR**
The information up to the point where the separator is missing is processed, but from that point until the next separator all information is lost.
- 12) **NUMERIC ATTRIBUTE—VAL REQD**
Non-numeric characters were input where a numeric value was expected.
- 13) **FILE NOT SPECIFIED**
An attempt has been made to modify an FPB without specifying FILE.
- 14) **PPB NOT SPECIFIED**
An attempt has been made to modify a PPB without specifying PPB.
- 15) **NOT AN INDEXED FILE**
An attempt has been made to modify an indexed file attribute for a non-indexed file.

Errors 1 to 4 only affect the one modification to which they refer. That particular modification is not processed, but all other correct ones are.

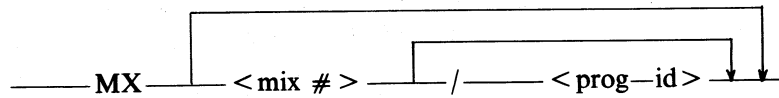
Errors 5 to 8 indicate that the specified modification has been effected, but that an inconsistency in the fields exists, and should be rectified.

Note that the use of this utility results in permanent modification of the object code file, and should therefore be used with care.

Note that the above messages are output by the utility. Any applicable system message may be output by the MCP (for example NO FILE if no printer is available).

MX (Diagnose Current Mix)

MX



This function, an intrinsic within the MCP, displays on the system journal the status of the specified task. If no task is specified, the status of all tasks currently in the mix is displayed.

If the requested task is not in the mix, the message

< INVALID MX >

is displayed,

If there are no tasks in the mix, the message

< NULL MIX >

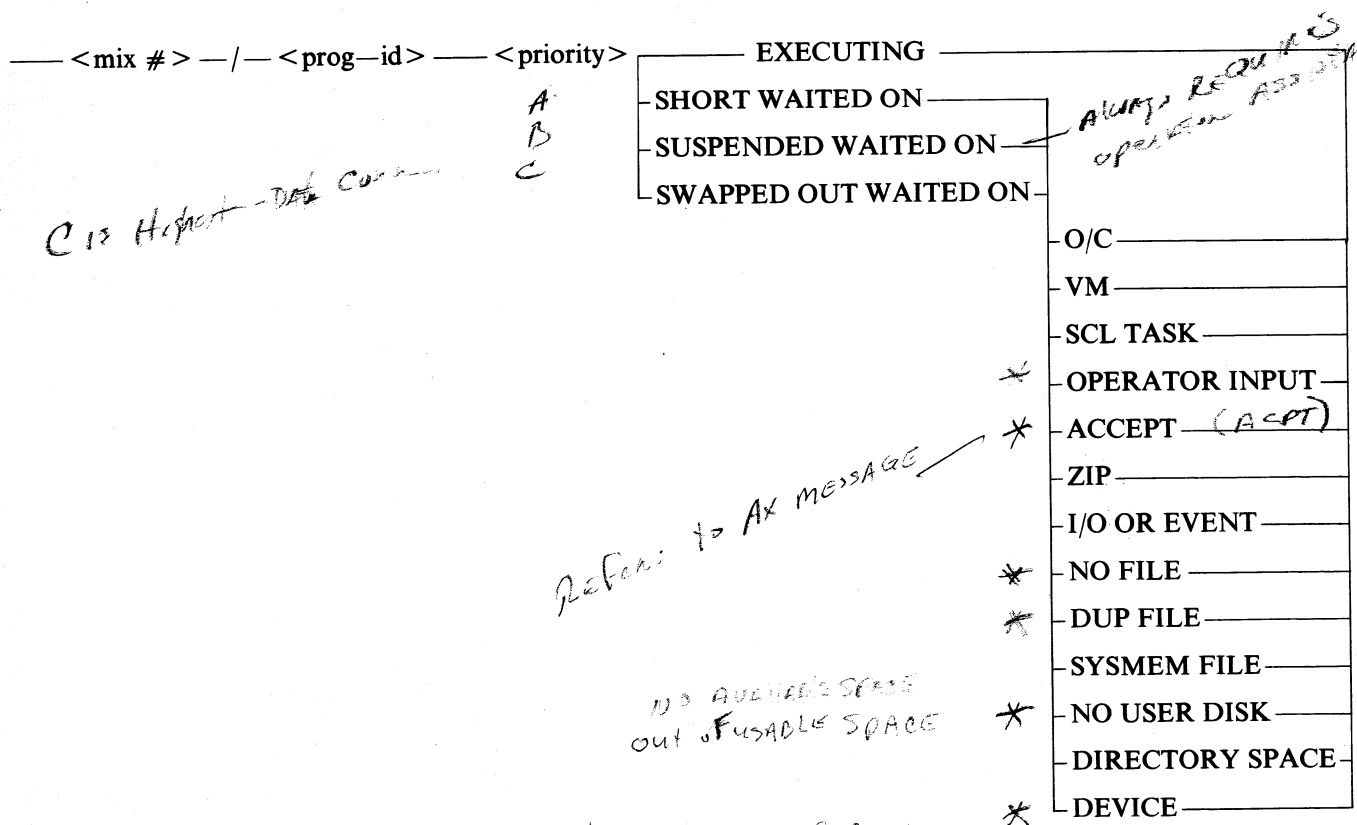
is displayed.

If the program-id specified does not correspond to the mix-number, the message

< INVALID PROGRAM ID >

is displayed.

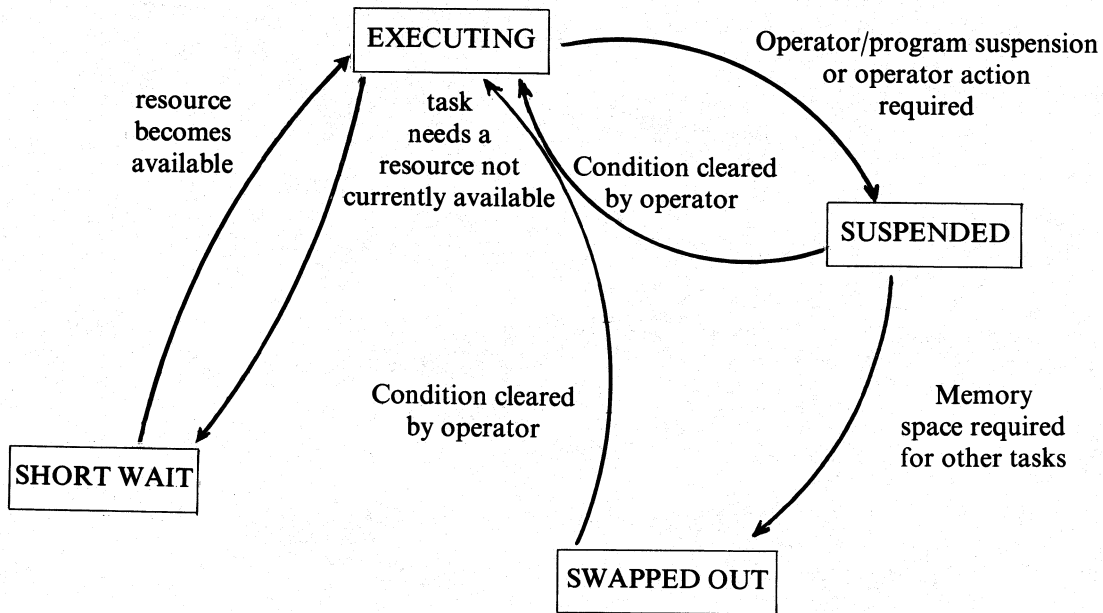
The information displayed for each task is described in the chart below:



Where < priority > is A, B or C indicating the class of the program.

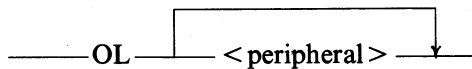
A task is **SHORT WAITED** if it requires a resource, such as Virtual Memory, or an I—O buffer, which the system can guarantee will be made available in a relatively short time. Any task waiting for operator action will be suspended. Suspended tasks are candidates for swapping to disk if their real memory space is required for other tasks in the mix.

Task States and Transitions



OL (Request for status information of a peripheral)

OL



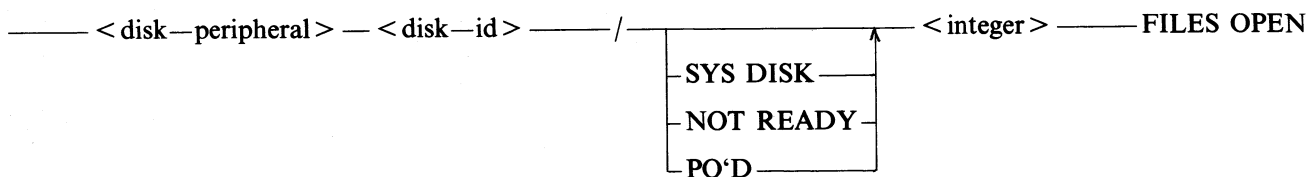
This function, an intrinsic within the MCP, allows the operator to request the status of peripherals on the system. If a peripheral is specified then the status of that peripheral is displayed, otherwise, the status of all peripherals on the system is displayed. A request for the status of an illegal or non-existent device will result in the message

OL <peripheral > INVALID

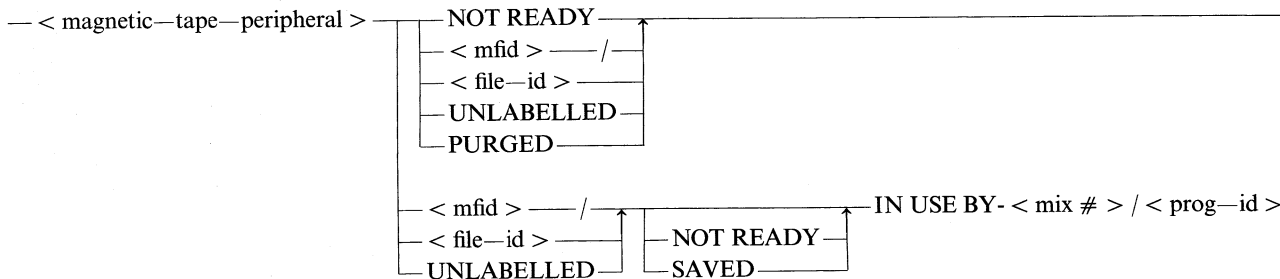
or OL <peripheral > NOT ON SYSTEM

being displayed.

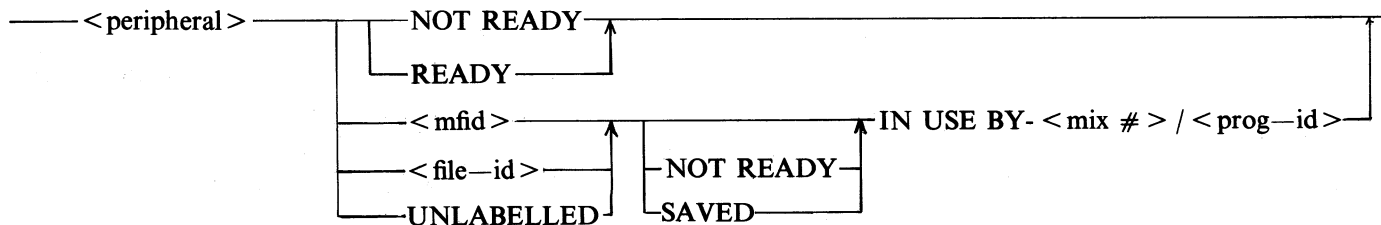
The message resulting from the OL command depends upon the type of peripheral and is summarized in the syntax charts below:



where <integer > specifies the number of files currently in use on the disk. For a magnetic-tape device.

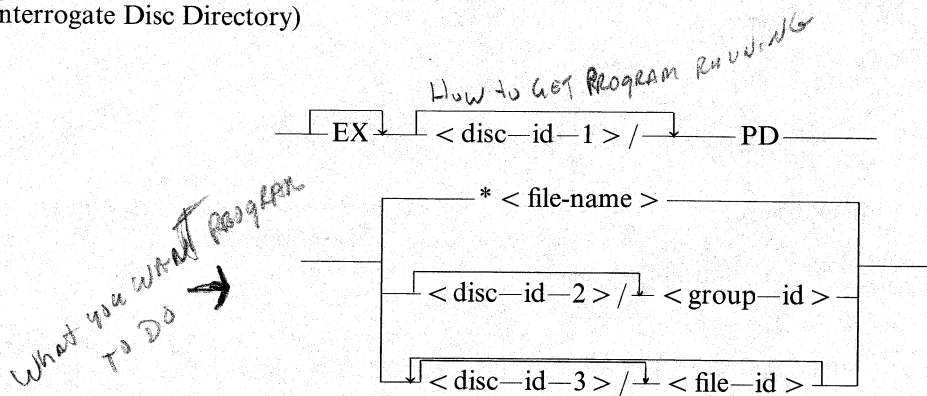


For any other device:



PD

PD (Interrogate Disc Directory)



This utility will verify the presence on—line of particular files or groups of files on disc. The utility may be executed from a disc other than the system disc by specifying `< disc-id-1 >`.

If the on—line presence of a group of files is to be verified, the particular group is identified by `< group-id >`. The group is assumed to be on the system disc unless `< disc-id-2 >` is specified. The utility will display a list of file—identifiers belonging to the group, or indicate that no such group has been found.

The on—line presence of specific files may be verified by specifying the `< file-id >` list. Each file is assumed to be on the system disc unless its respective `< disc-id-3 >` is specified. The utility will display a list of file—identifiers found to be on-line, and a list of file—identifiers which cannot be found.

Output Format

If the `< group-id >` has been specified and files are found which belong to that group, then

`< group-id > ON < disc-id-2 > CONTAINS:—`

is displayed on the system journal (SPO) followed by the file—identifiers of the files contained in the group, three identifiers per line. The list is completed with the message

END PD.

If the `< group-id >` has been specified and no relevant files are found, then the message

NO FILES FOUND IN THE FAMILY `< group-id >`

is displayed on the SPO, followed by the message

END PD.

If a list of files has been specified, then for those files specified in the parameters list which have been found the message

ON LINE

is displayed on the SPO, followed by the file—ids of the located files, two identifiers per line. If any of the files specified in the parameter list are not found, then the message

NOT ON LINE

is displayed on the SPO, followed by the identifiers of those files which have not been located. After all the specified files have been processed, the message

END PD

is displayed.

Output Messages

The following message may be displayed by the utility in the event of the corresponding error or condition.

Errors in Initial Parameters

The following messages will be displayed, and the utility will terminate, if errors are found in the initial parameters following PD in the SCL input string.

1) **ILLEGAL PARAMETER LIST**

This message is displayed if the initial parameters are incorrect. The utility will attempt to follow the message with a number of characters from the area of the input message which caused the failure.

2) **INVALID CHARACTER IN IDENTIFIER < identifier >**

This message is displayed if the < disc-id >, < group-id >, or < file-id > entries in the input message contain characters which are illegal in identifiers.

3) **< file-name > NOT FOUND**

This message is displayed if the utility cannot locate the parameter—file identified in an * < file-name > macro-call.

4) **< file-name > REQUIRES OVERFLOW DISK < disc-id >**

This message is displayed if a file which extended over two discs has only the base disc on line.

5) **NO SPECIFICATION GIVEN**

This message is displayed if no parameters are entered after PD in the SCL input string.

Errors During Execution

The following messages will be displayed, and the utility will proceed to the next request in the parameter string, if the corresponding error is encountered during execution of the utility.

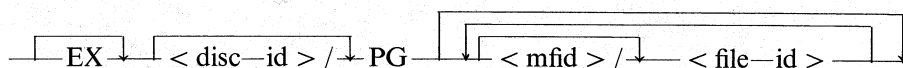
1) **PACK < disc-id > NOT OPENED—NOT ON LINE**

This message is displayed if the identified disc cannot be located. All subsequent references to this disc in the input parameters will be ignored.

Note that the above messages are output by the utility. Any applicable system message may be output by the MCP (for example ERROR WHILE IN < verb > if disc parity errors are encountered).

PG

PG (Purge Tape)



This utility provides for the purging (labelling as available for output) of magnetic tape and cassette files. The utility may be executed from a disc other than the system disc by specifying < disc-id >.

A list of labelled tape < tape-names > may be input, in which case the < mfid > and < file-id > of each < tape-name > specified must exactly correspond to the first label of each tape (that is, a multifile tape must have the < mfid > and < file-id > of the first file correctly identified). For example, PG MEMDUMP/MEMORY.

One unlabelled tape may be purged in each execution of PG by entering no parameters in the input message. In this case, the utility may be directed to the required tape by use of the AD intrinsic. For example

```
< input > PG
< output > 10/PG [14] WAITING UNLAB TAPE MT DEVICE REQUIRED
< input > AD 10/PG CSA
```

This method may be used to purge a labelled tape for which the exact < tape-name > is not known.

Output Format

The following messages may be output by the utility during execution.

- 1) < file-name > HAS BEEN PURGED
This message is displayed for each file which is purged. The utility will proceed with the next file in the file-name list.
- 2) < file-name > NOT ON LINE
The specified file has not been located. The utility will proceed with the next file in the file-name list.
- 3) < file-name > NO WRITE PERMIT
A request has been made to purge a file which has been found, but which has no write capability. The utility will proceed with the next file in the file-name list.
- 4) UNLABELLED TAPE—NO WRITE PERMIT
As (3) above, but when an unlabelled tape has been specified. The utility will terminate.
- 5) UNLABELLED TAPE—HAS BEEN PURGED
As (1) above, but when an unlabelled tape has been specified. The utility will terminate.
- 6) < file-name > NOT FOUND
This message is displayed if the file containing the input parameters (* file) cannot be found.

Errors in Initial Parameters

The following messages will be displayed and the utility will terminate, if errors are found in the initial parameters following PG in the SCL input message.

- 1) ILLEGAL PARAMETER LIST
This message is displayed if the initial parameters are incorrect. The utility will attempt to follow the message with a number of characters from the area of the input message which caused the failure.
- 2) INVALID CHARACTER IN IDENTIFIER < identifier >
This message is displayed if the < mfid > or < file-id > entries in the input message contain characters which are illegal in identifiers.

Note that the above messages are output by the utility. Any applicable system message may be output by the MCP (for example ERROR WHILE IN < verb > if tape write errors are encountered).

PMB80 (Analyse B80 RAM Dump)

PMB80

This utility performs selective and comprehensive analysis of the cassette tapes MEMDUMP/MEMORY which are output by the Warmstart Memory Dump facility on B80. The utility is not applicable to other CMS implementations. Operation of the utility requires a full understanding of the information presented in the CMS MCP Reference Manual, form number 2007555, therefore details of the utility are provided in the B80 dependent section of that manual. Two parameter files are provided for use by the utility. These files are identified by the prefix "PM" in their file—ids.

PO

PO (Power off a disk drive)

—— PO —— < disk peripheral > ——

This MCP intrinsic allows the operator to logically power off a disk drive. If no files on the disk specified are in use by any program, the disk is logically powered off and the message

< disk peripheral > OK

is printed.

If files on the specified disk are in use, the OL message for the disk is printed. No further task will be allowed to open files on the disk and when all files in use have been closed, the disk will be logically powered off and

< disk peripheral > OK

will be printed.

A disk should not be physically powered off until it has been logically powered off otherwise information being written to the disk may be lost. If a disk is physically powered off before a logical PO then the message

< disk peripheral > REMOVED WITHOUT PO

is printed. Any program using files on that disk will eventually terminate with an error condition indicating hardware failure.

A PO'd may be made ready again by the RY command or by physically powering the unit off and on.

If the specified unit is invalid or not on the system, the messages

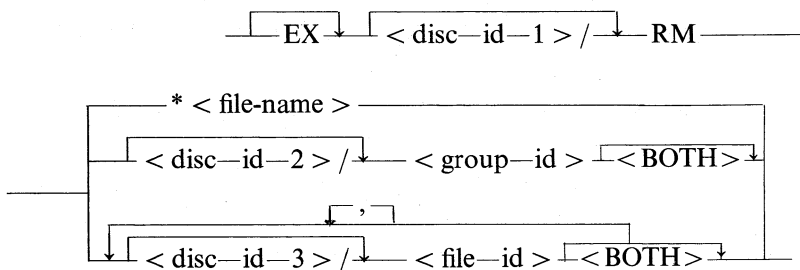
PO < disk peripheral > INVALID
or PO < disk peripheral > NOT ON SYSTEM

will be printed.

If the disc specified is the system disc, then the command is taken to be a command to power off the system. Invalid attempts to power off the system result in the message

CANNOT POWER OFF SYSTEM. MIX NOT EMPTY

An invalid attempt to power off the system has no consequence beyond the printing of the message.



This utility provides for the removal of files from disc. The utility may be executed from a disc other than the system disc by specifying < disc-id-1 >.

If a group of files is specified for removal, the group is assumed to be on the system disc unless < disc-id-2 > is specified. The required group is identified by < group-id >. If the group is not found then the message NO FILES FOUND FOR REMOVAL IN THE FAMILY < group-id > is displayed.

A list of files, not necessarily on the same disc, may be specified. Each file identified in the < file-id > list is assumed to be on the system disc unless its respective < disc-id-3 > is specified.

The disc areas associated with those files which are found and which can be removed are returned to the available table. A request for the removal of system files causes the utility to output:

< disc-id > / < file-id > IS A SYSTEM FILE

The utility will then issue an ACCEPT to which the response

AX < mix-number > /RM < disc-id > / < file-id > -OK

will cause the removal of the identified file.

Note that the file must be identified in the AX exactly as identified by the utility. For example, if the file is known to be on the system disc, the AX must still include < disc-id > if the utility message specifies < disc-id >. Any response other than that shown above will be treated as a request not to remove the system file, and the utility will continue as if the request to remove the file had not been made.

The utility will remove both files of an indexed pair if the relevant keyword is included in the parameter list. The keyword "< BOTH >" may be specified after any < file-id > in the file-name list, or after < group-id >. If RM detects that a Key File is being removed and < BOTH > has been specified then it will remove both the Key File and the associated Data File if both are on-line. If neither, or only one is on-line then neither the Key File nor the Data File will be removed.

Output Messages

The following messages will be output by the utility after a request has been processed. The utility will then continue with the next request, or terminate if the list of requests is exhausted.

- 1) < file-name > REMOVED
This message is displayed for each file removed.

- 2) < file—name > NOT REMOVED—NOT FOUND
This message is displayed if the identified file is off-line.
- 3) < file—name > NOT REMOVED—IN USE
This message is displayed if a request is made for the removal of a file which is in use at the time of the request.
- 4) < file—name > NOT REMOVED—SYSTEM FILE
This message is displayed if the response to a request to remove a system file was not < file—name > OK.
- 5) < file—name > AND < file—name > REMOVED
This message is displayed for each pair of Key and Data files removed.
- 6) NO SPECIFICATION GIVEN
This message is displayed if no parameters are entered following RM in the SCL input string.

The following messages are displayed, and the utility terminates, in the event of the corresponding error or condition.

- 1) ILLEGAL PARAMETER LIST
This message is displayed if the input message following RM in the SCL input string is incorrect. The utility will attempt to follow the message with a number of characters from the area of the input message which caused the failure.
- 2) INVALID CHARACTER IN IDENTIFIER < identifier >
This message is displayed if the < disc—id > or < file—id > entries in the input parameters contain characters which are illegal in identifiers.
- 3) < file—name > NOT FOUND
This message is displayed if the parameter file identified by an * < file—name > macro—call cannot be located.
- 4) DISK < disc—id > NOT OPENED—NOT ON LINE
This message is displayed if the identified disc cannot be located.

Note that the above messages are output by the utility. Any applicable system message may be output by the MCP (for example ERROR WHILE IN < verb > if a disc containing files for removal has been illegally powered—off.)

RY (Ready a peripheral)

RY

—— RY —— < peripheral > ——

This intrinsic is used to ready a peripheral that has been made logically not ready, to enable it to be assigned for future use. The RY command may be used to ready a PO'd disk, SV'd printer or tape unit, or any peripheral that has programmatically been closed with lock. If a peripheral is awaiting a PO or SV command because it is still in use then the RY command will cancel the previous SV or PO.

If the operator attempts to ready a peripheral that is physically not ready then the appropriate OL message for that peripheral is printed.

An attempt to ready an illegal peripheral or a peripheral not on the system, then the messages

RY < peripheral > INVALID

or RY <peripheral > NOT ON SYSTEM

are printed.

ST

ST (Temporarily Suspend a Running Task)

———ST——— <mix #> ———/——— <prog-id> ———

This intrinsic causes all real store currently in use by the task to be removed to the virtual memory disk space allocated for this task. The task still appears in the mix and all peripheral assignments are maintained. All real store thus freed is released for re-allocation. If the option < prog-id > is specified, then the mix number and prog-id must match, otherwise, the invalid stop message is displayed.

If the stop is accepted, the message

<mix #> / <prog-id> STOPPED

is displayed. If the specified task does not exist or is already stopped, then the message

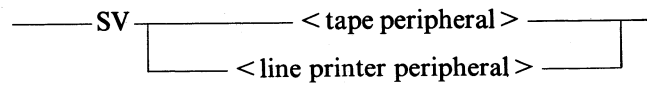
<mix #> / <prog-id> ST INVALID

is displayed.

The task may be restarted by the GO command.

SV (Save Peripheral)

SV



This MCP intrinsic allows the operator to logically power off a line printer or a tape unit in order to prevent its use by any program.

If no program is using the specified device, it is marked as being logically not ready and the message
< peripheral > OK
is printed.

If the device is in use, the OL message for that peripheral is output. When the device is released by the program, it is marked as being logically not ready and the message

< peripheral > OK

is printed.

A saved device may be made ready again by the RY command or by physically powering the unit off and on.

If the specified peripheral is invalid or non-existent on the system, one of the messages

SV < peripheral > INVALID

or SV < peripheral > NOT ON SYSTEM

will be printed.

TAPELR

TAPELR (List Library Tape Directory)

— EX — < disc-id > / — TAPELR — < library-tape-name > —

This utility will produce a listing on a line printer (or the console printer if no line printer is available) of the attributes of files contained on library tapes. A library tape is a tape produced by the DUMP or UNLOAD functions of the LD utility. The utility may be executed from a disc other than the system disc by specifying < disc-id >.

One or more library tapes may be analysed during one execution of the utility. The required tapes are identified by specifying a < library-tape-name > list. For each tape identified in the input specifications the following attributes are analysed:

Number of Records in the file
Record Size
Block Size
Creation Date
Last Access Date
File Type
Area Sizes

Output Format

Eight columns of information will be output to the printer for each library tape for which information is requested. The column headings, the format of the values which these columns contain, and the significance of these values is as follows:

<u>Heading</u>	<u>Value</u>	<u>Significance</u>
FILE NAME	12 characters	The File identifier
ACTUAL SIZE	7 digits	The number of records in the file
MAXIMUM SIZE	7 digits	The maximum number of records which may be written to the file
RECORD SIZE	5 digits	The number of characters in the record
RECORDS PER BLOCK	5 digits	The number of records in each block
CREATION DATE	5 digits	File Creation Date (Julian, YYDDD)
LAST ACCESS DATE	5 digits	Last Access Date (Julian, YYDDD)
FILE TYPE	8 characters	See Note Below

Note. The FILE TYPE entry will be one of:

DATA	—	Normal Data File
CODE	—	S-code File
KEY	—	Key File
SYSTEM	—	System File (interpreter, etc.)
SRCELANG	—	Source Language File
SRCELIBR	—	Source Library File

Output Messages

The following messages will be displayed by the utility in the event of the corresponding error.

Errors in Initial Parameters

The following messages are displayed, and the utility will terminate, if errors or inconsistencies are detected in the input parameters.

1) ILLEGAL PARAMETER LIST

This message is displayed if the input parameters following LR in the SCL input message are incorrect. The utility will attempt to follow the message with a number of characters from area of the input message which caused the failure.

2) INVALID CHARACTER IN IDENTIFIER < identifier >

This message is displayed if any of the < library—tape—names > specified contain characters which are illegal in identifiers.

3) < file—name > NOT FOUND

This message is displayed if the parameter—file identified in an * < file—name > macro-call cannot be located.

4) NO SPECIFICATION GIVEN

This message is displayed if no parameters are entered following TAPELR in the SCL input string.

Errors During Execution

The following message is displayed if a specified < library—tape—name > is of an incorrect format.

< library—tape—name > NOT A RECOGNISED DUMP TAPE

The utility will ignore this tape and process any further requests in the specification string.

Note that the above messages are output by the utility. Any applicable system message may be output by the MCP (for example NO FILE, if an identified < library—tape > cannot be found).

TAPEPD

TAPEPD (Interrogate Library Tape Directory)



This utility will display, upon the console, a list of the files dumped on specified library-tapes. A library tape is a tape produced by the DUMP or UNLOAD functions of the LD utility. The utility may be executed from a disc other than the system disc by specifying <disc-id>.

One or more library tapes may be interrogated during one execution of TAPEPD. The required tapes are identified by the <library-tape-name> list.

Output Format

For each library tape identified in the input parameters, the following information is displayed:

MT <library-tape-name> DUMPED ON <day-of-week> <DD> <month> <YY> CONTAINS:—

This message precedes the list of files found on each tape. The <day-of-week> and <month> entries are three character abbreviations. The <DD> and <YY> entries are two digit entries showing the day of the month and the year respectively. The message is followed by the list of files, three files per line. At the end of the last list of files, the message

END TAPEPD

is displayed.

Output Messages

The following messages are output by the utility in the event of the corresponding error or condition.

Errors in Initial Parameters

The following messages are displayed, and the utility will terminate, if errors or inconsistencies are detected in the initial parameters following TAPEPD in the SCL input message.

1) ILLEGAL PARAMETER LIST

This message is displayed if the initial parameters are incorrect. The utility will attempt to follow the message with a number of characters from the area of the input message which caused the failure.

2) INVALID CHARACTER IN IDENTIFIER <identifier>

This message is displayed if any <library-tape-name> contains characters which are illegal in identifiers.

3) <file-name> NOT FOUND

This message is displayed if the parameter-file identified by an * <file-name> macro-call cannot be located.

4) NO SPECIFICATION GIVEN

This message is displayed if no parameters are entered following TAPEPD in the SCL input string.

Errors During Execution

The following message is displayed if a specified <library-tape-name> is of an incorrect format.

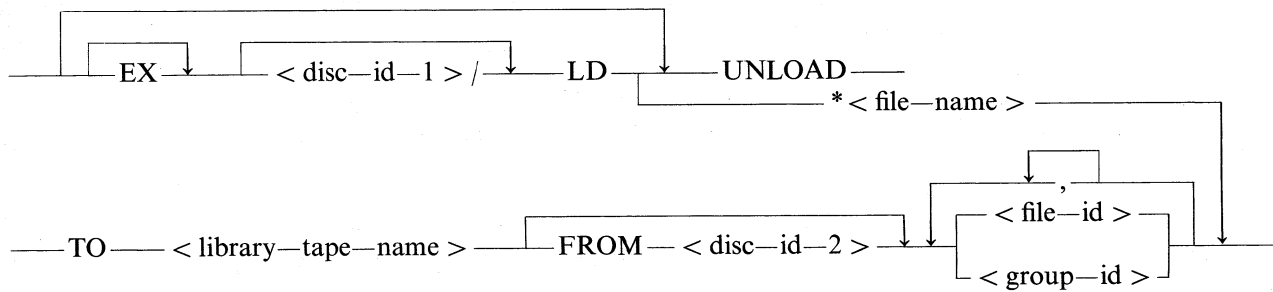
<library-tape-name> NOT A RECOGNISED DUMP TAPE

The utility will ignore this tape and process any further request in the specification string.

Note that the above messages are output by the utility. Any applicable system message may be output by the MCP (for example, NO FILE if an identified <library-tape> cannot be located).

UNLOAD (Unload files to Library Tape)

UNLOAD



Note that this function is a sub-program within the utility LD. MCP recognises the mnemonic UNLOAD if LD is not specified, and will automatically initiate a load of the LD utility. The < file-id > of LD should not therefore be changed if the UNLOAD function is to be invoked in this manner. To discontinue the function, "DS < mix-number > /LD" must be used.

This function may be executed from a disc other than the system disc by specifying LD and < disc-id-1 >.

The function will create a library tape of name < library-tape-name > containing files copied from the disc specified by < disc-id-2 > (or the system disc if < disc-id-2 > is omitted). Each disc file copied to tape is deleted from the disc after the file has been copied. The file list identifies the particular files or groups of files to be copied to the library tape.

Note that the file list may contain a mixture of files and groups of files.

Output Messages

The following messages will be output by the utility in the event of the corresponding errors or conditions.

Errors in Initial Parameters

The following messages are output, and the utility will terminate, if the initial parameters are incorrect or invalid.

1) ILLEGAL PARAMETER LIST

This message is displayed if the parameters supplied in the input message are incorrect. The utility will attempt to follow the message with a number of characters from the area of the input message which caused the failure.

2) INVALID CHARACTER IN IDENTIFIER < identifier >

This message is displayed if the < library-tape-name >, < disc-id >, or < file-id > entries in the input message contain characters which are illegal in identifiers.

3) < file-name > NOT FOUND

This message is displayed if the parameter-file identified by an * < file-name > macro-call cannot be located.

4) DISK < disc-id > NOT AVAILABLE

This message is displayed if the specified disc is not on line.

5) NO SPECIFICATION GIVEN

This message is displayed if no parameters are entered following LD in the SCL input string.

Errors During Execution

The following messages are displayed if the utility cannot perform some part of its function. The utility will continue to execute wherever possible.

1) NO FILES IN THE FAMILY < group—id > ON DISK < disc—id > FOR UNLOAD

This message is displayed if the utility is unable to find any files in the specified group on the disc. The utility will continue with the next file in the input parameter list.

2) NO FILE < file—id > ON DISK < disc—id > FOR UNLOAD

This message is displayed if a specified file is not present on the disc. The utility will continue with the next file in the input parameter list.

3) < file—name > NOT DUMPED—IN OUTPUT USE

This message will be displayed if a particular file is found to be in use. The tape will be purged and the utility will terminate.

4) < file—name > NOT DUMPED—HAS BEEN REMOVED

This message is displayed if a file is removed between the start of UNLOAD and the time when the file is to be copied to tape. The tape will be purged and the utility will terminate.

5) < file—name > NOT DUMPED—HAS BEEN ALTERED

This message is displayed if the contents of a file are changed between the start of UNLOAD and the time when the file is to be copied to tape. The tape will be purged and the utility will terminate.

6) < file—name > LOAD/DUMP DISCREPANCY

This message will be displayed if end of file has been reached before it is expected, and implies erroneous information in the Disk File Header.

Normal Execution

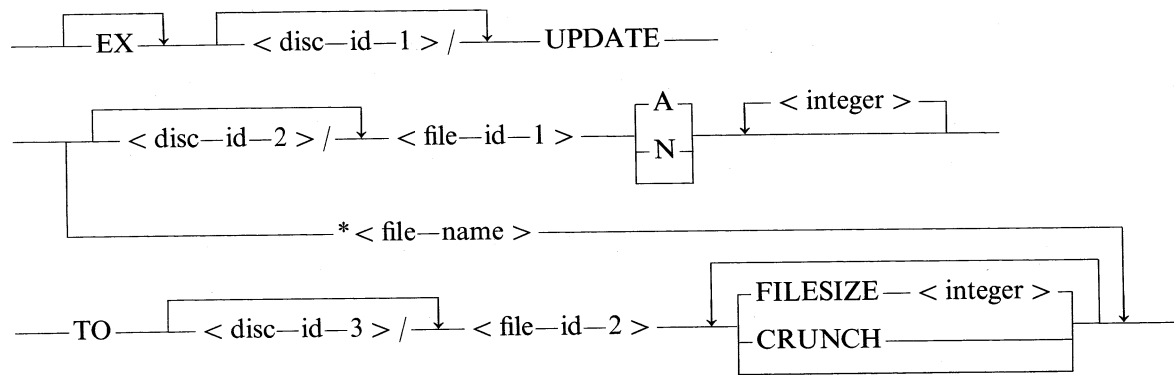
The following message is displayed for each file removed.

< file—name > REMOVED

The following message is displayed for each file which is copied to tape.

< file—name > DUMPED

Note that the above messages are output by the utility. Any applicable system message may be output by the MCP (for example, ERROR WHILE IN < verb > if the tape cassette drive is opened while in use).



This utility may be used to construct new disc files from existing files and information specified through the console keyboard. The utility may be executed from a disc other than the system disc by specifying `< disc-id-1 >`. The utility may only be used with systems incorporating a console keyboard.

The existing file is identified by `< file-id-1 >` and is assumed to be on the system disc if `< disc-id-2 >` is omitted. The file must be of type Source or type Data, and the necessary attributes, such as Record Size, will be taken from this file. For files of type Data, the input format may be specified to be alphanumeric (A) or hexadecimal (N). The default if no specification is given is alphanumeric (A). Source file input and Data file type A input is accepted as direct keyboard input, whereas type N will require the input of two characters (0-9, A-F) for each byte of the record.

The integer list specified in the syntax may be used to provide "tab" positions within the record. The use of OCK1 when keying input data causes the utility to reposition the input point in the record to the next tab position. During this repositioning, the utility will fill all character positions left unspecified in the record with a fill character determined by the input type. For source input, the fill character will be an ASCII space, for alphanumeric input, an ASCII zero, and for hexadecimal input, a binary-zero-filled character. The record length plus one is used as a terminating tab position (whether or not other tabs are specified).

The utility can be used for record sizes up to 500 bytes, but, since the utility cannot be given input greater than the width of the console, tab positions are mandatory on files of larger record sizes. For example, a file of 180 byte records requiring alphanumeric input will require at least one tab position (for instance at position 100), whereas a file of 180 byte records requiring hexadecimal input will require a minimum of two tab positions (for instance at positions 60 and 120).

The new file to be created is identified by `< file-id-2 >`. The file will be output to the system disc unless `< disc-id-3 >` is specified. The maximum number of records likely to be written to the new file may be specified using the FILESIZE clause. The `< integer >` specifies the total number of records, and will be taken from the old file if not specified. The CRUNCH clause allows the new file to be closed with the crunch flag set. The new file will occupy the minimum area of disc, but can never be extended.

The utility operates in three modes—Record Modify Mode (PK2), Record Select Mode (PK3), or Record Insert Mode (PK4). The PK associated with the currently active mode is disabled (the light is turned off) to indicate the appropriate mode. PK1, PK5, and PK6 are enabled at appropriate points in each mode to:—

- a) PK1. Write the last record processed to the new file then select and print the next logical record from the old file. The printout will show the record number in the old file of the selected record, together with the next record number to be written to the new file.
- b) PK5. Delete the last record printed by selecting and printing the next logical record from the old file without writing the last record to the new file. The printout will show the record numbers in the old file of the selected record, together with the next record number to be written to the new file.

c) PK6. Terminate the utility. All records in the old file not processed will be copied to the new file, and all files will be closed.

When execution of the utility begins, Record Select Mode is automatically entered.

PK3 Record Select Mode

The required record is identified by logical record number using the following syntax

— RECORD — < integer > —

where < integer > cannot be less than the number of the last record obtained from the old file, or greater than the number of records in the file. During the process of locating the required record, all records from, and including, the last record processed, up to the one immediately prior to the selected record, will be copied from the existing file to the new file. The record selected, when found, will be printed with its record number in the old file followed by the record number that the next record written to the new file will take. The print format of the record contents will correspond to the input mode selected, that is alphanumeric, source, or hexadecimal. Record Modify Mode or Record Insert Mode may then be selected. Note that a record inserted by Record Insert Mode will be positioned *after* the selected record in the new file. Selecting Record 0 allows records to be inserted before Record 1 of the old file.

PK2 Record Modify Mode

The point in the record at which alterations are to be made is selected by presenting an identifying character string which immediately precedes the required byte of the record. The character string for insertion or replacement follows the identifying string, delimited by colons (:). If alterations are to be made at the beginning of the record, no identifying string is input. The syntax of the keyboard input is:—

| < identifying—string > ↓ : < modifying—string > :

The use of OCK1 or OCK2 to terminate the input determines whether the < modifying—string > will replace or be added to the existing characters in the record.

OCK1—Replacement

The < modifying—string > replaces the corresponding number of characters in the record if OCK1 is used to terminate the input. For example, with a record containing

ABCD0123

the amendment C:XY: would result in a record containing

ABCXY123.

OCK2—Insertion

The < modifying—string > is inserted at the indicated point if OCK2 is used to terminate the input. The insertion can cause characters in the record to be moved to the right. The shifting of characters applies only to those characters from the starting byte to the next higher relevant tab position; characters beyond this tab position will not be affected.

For example, with a record containing

ABCDEFGH1234

the amendment C:WXYZ: would result in the record

ABCWXYZD1234 if the tabs specified were 7 and 9, or the record

ABCWXYZDEFGH

if the tab specified was 7 alone.

On completion of a modification, the utility will print the amended record with its associated record number, and then illuminate all other usable PK options for possible selection.

PK4 Record Insert Mode

This mode allows additional records to be inserted in the new file after the last selected record of the old file. The keyboard input format is determined by the mode specified in the input message, that is alphanumeric or hexadecimal with appropriate character fill when required. The input must be made in accordance with the specified tab stops.

The utility prints the record number in the old file of the last record taken from the old file, and the record number in the new file of the next record to be output, prior to accepting keyboard input. The input is echo-printed. When all insertions have been made at a particular point in the file, an available PK may be pressed to select the next mode or terminate the utility. Note that to insert a record at the beginning of the new file, Record 0 should be selected in Record Select Mode, prior to selecting Record Insert Mode.

Output Messages

The following messages will be output by the utility in the event of the appropriate error.

Errors in Initial Parameters

These errors occur if the initial parameters following "UPDATE" in the SCL input string are incorrect. After the message is displayed, the utility terminates

- 1) FILETYPE IS NOT SOURCE OR DATA
This message is displayed if the file identified as the old file is of an incorrect file-type.
- 2) < file—name > NOT FOUND
This message is displayed if the file identified as the old file, or the parameter—file identified by the * < file—name > macro-call, cannot be located.
- 3) ILLEGAL PARAMETER LIST—ATTRIBUTE SPECIFICATION INVALID
This message is displayed if there is an incompatibility between the specified record and block sizes.
- 4) ILLEGAL PARAMETER LIST—TABS ERROR
This message is displayed if tab positions beyond the end of the record are specified, or imply input fields larger than the capability of the console.
- 5) ILLEGAL PARAMETER LIST
The utility will attempt to follow this message with a number of characters from the area of the input message which caused the failure.
- 6) INVALID CHARACTER IN IDENTIFIER < identifier >
This message is displayed if the < disk—id > or < file—id > portions of the file names contain characters which are illegal in file—names.

7) NO SPECIFICATION GIVEN

This message is displayed if no parameters are entered following UPDATE in the SCL input string.

Errors During Execution

The following errors may be encountered during the execution of the utility, and do not cause the utility to terminate. The messages are displayed via the console file.

1) NOT HEXADECIMAL CHARACTER INPUT—RESUBMIT

If the input mode is hexadecimal, any character other than 0–9 and A–F will cause the above message to be displayed. The complete entry must be resubmitted.

2) ODD NUMBER OF HEXADECIMAL CHARACTERS INPUT

If the input mode is hexadecimal, and the number of hexadecimal characters is odd, then this warning is displayed. The utility processes the hexadecimal string by extending it on the right with a zero character.

3) INPUT ERROR—RESUBMIT RECORD MODIFICATION

If the syntax of the keyboard input in Record Modify Mode is incorrect, the above message is displayed. The entire entry is discarded.

4) BYTE WITHIN RECORD SPECIFIED NOT FOUND

The < identifying—string > in the keyboard input in Record Modify Mode could not be found in the record. The entire entry is discarded.

5) RECORD SELECTION ERROR

This message is displayed if, when in Record Select Mode, an invalid record request is submitted.

6) UNWANTED KEY PRESSED—PLEASE RE-INPUT

This message is displayed if an unexpected terminating key is used. The entry should be re-submitted.

7) INPUT IMMEDIATELY BEFORE PK6 HAS BEEN LOST

This warning message is displayed if characters were input immediately before PK6 was pressed to terminate the utility. The characters will not be written to the output file, and the utility will terminate normally.

The following errors may be encountered during the execution of the utility, and will cause a normal termination (that is, identical to selecting PK6) of the utility.

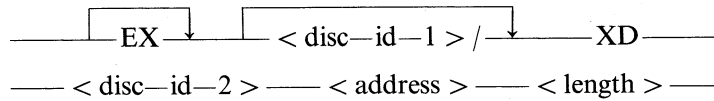
1) RECORD REQUESTED IS BEYOND E.O.F.

This message is displayed if, when in Record Select Mode, a request is made for a record beyond the end of the old file.

2) E.O.F. REACHED DURING DELETIONS

This message is displayed if, when in record delete mode, a request is made to delete a record beyond the end of the old file.

Note that the above messages are output by the utility. Any applicable system message may be output by the MCP (for example DUPLICATE FILE if the new file has the same name as a file already present on the disc).



This utility will allow the removal of contiguous disc sectors from the Available Table of the required disc. Once the sectors are removed, they may only be restored to use following a successful disc initialisation, so care must be exercised in the use of this utility. The utility should be executed as the only task in the mix, and should not be executed from the disc from which it is intended to remove sectors.

The utility may be executed from a disc other than the system disc by specifying < disc-id-1 >. The disc from which sectors are to be deleted is identified by < disc-id-2 >. The utility will delete one contiguous block of sectors in each execution of the utility. The disc address of the first sector to be deleted is specified, in hexadecimal, in the < address > entry. The number of sectors to be deleted from that point is specified, in hexadecimal, in the < length > entry. Note that "@" characters are not required to delimit these fields.

In order to remove the disc space, the relevant sectors must be in the available table.

Identification of Bad Sectors

The appearance of bad sectors on a particular disc will be evidenced by recurrent errors of the following type whilst using the disc:

< mix-number > / < program-id > [< event-number >] < file-name > DK < function >
 < message > ERROR WHILE IN < verb > < status >

where < event-number > and < message > will be as shown below.

< event-number >	< message >
2	PARITY
3	TIMEOUT
4	ADDRESS
45	PARITY
46	TIMEOUT
47	ADDRESS

} Program will continue to execute

} Program will require DS or DP

For the above events, < status > will show the appropriate disc sector address (PARITY and ADDRESS) or disc cylinder address (TIMEOUT) in hexadecimal.

Important Note

This utility is a task which runs under MCP control, and as it is a utility which writes records to the disc directory, it will not be able to exercise a great deal of control over tasks attempting to, for example, acquire disc space at the same time. XD should therefore be run as the only task in the mix, and should not be executed from the disc from which it is intended to remove sectors if this can possibly be avoided.

Output Messages

The following messages will be output by the utility in the event of the corresponding error or condition.

Errors in Initial Parameters

The following messages will be displayed, and the utility will terminate, if errors are detected in the initial parameters following "XD" in the SCL input message.

- 1) < parameter—list > ILLEGAL PARAMETER LIST
This message is displayed if the parameters are found to be unacceptable.
- 2) DISK < disc—id > FOR XD NOT AVAILABLE
This message is displayed if the specified disc is not on line.
- 3) INVALID CHARACTER IN IDENTIFIER < identifier >
This message is displayed if the character string representing < disc—id—2 > contains characters which are invalid in identifiers.
- 4) NO SPECIFICATION GIVEN
This message is displayed if no parameters are entered following XD in the SCL input string.

Errors During Execution

The following messages will be displayed if the utility cannot entirely perform the requested deletion.

- 1) ONLY @ < length > @ SECTORS CAN BE DELETED
This message is displayed if the number of sectors specified for deletion is greater than the number contained in the Available Table entry. The utility will proceed with the reduced number indicated by @ < length > @.
- 2) AVAILABLE TABLE FULL—ENTRY @ < address > @ @ < length > @ LOST
This message is displayed if, following the removal of the requested sectors, there are no entries left in the available table for the inclusion of the zero, one, or two new available areas created. The utility will proceed otherwise normally to end of job. The disc concerned may still be used for running tasks, but there will be sectors which cannot be accessed by any task (these may be shown by a KA listing of the disc); they may only be retrieved by initialising the disc.
- 3) SECTORS FOR XD NOT IN AVAILABLE TABLE
This message is displayed if the sectors requested for removal are allocated to a file, previously removed, or missing. The utility will terminate.

Normal Execution

The normal, successful, termination message is

@ < length > @ SECTORS FROM @ < address > @ DELETED

Note that the above messages are output by the utility. Any applicable system message may be output by the MCP (for example, ERROR WHILE IN < verb > if the directory structure of the target disc contains bad sectors).

*USED BY
PROGRAMMER*

SECTION 4 SORT/MERGE INTRINSICS

INTRODUCTION

This Section is intended as a comprehensive description of the capabilities, the requirements, and the visible interfaces of the CMS sort-merge intrinsics.

Syntactic and semantic definitions of sort and merge constructs appearing in applications languages are not discussed in detail (although the CMS COBOL sort and merge verbs are briefly introduced). Refer to the relevant language manuals for precise definitions.

The sort language (used to initiate a 'stand-alone' sort or merge) is defined in detail.

RELATED DOCUMENTS

2007555 CMS Master Control Program (MCP) Reference Manual.
2007266 CMS COBOL Reference Manual.
2007274 CMS RPG Reference Manual.

GENERAL DESCRIPTION

The CMS sort-merge intrinsics provide the following capabilities:

- a) The records contained within a designated data file may be sorted on a series of specified keys, each key ascending or descending as required.
- b) A tagfile (suitable for use as an ADDROUT file in RPG and for limited indexed access in COBOL) may be created from a designated data file using a series of specified keys, each key ascending or descending as required.
- c) A keyfile (suitable for full CMS indexed access) may be created from a designated data file using a specified unsigned key (ordered ascending). Optionally, a check for duplicate key values may be requested.
- d) A number of designated files may be merged together using a series of specified keys, each key ordered ascending or descending as required.

Any of the above functions may be invoked using the sort language (that is, by executing a stand-alone sort or merge).

Sort and merge constructs found in applications languages are in general defined such that only a subset of the above functions may be invoked. The facilities available in CMS COBOL are outlined in COBOL GENERATED SORTS AND MERGES; however, for both COBOL and other CMS application languages, consult the appropriate language manual for a precise definition of the functions to which access may be made.

Function a) listed above can be performed using a minimal amount of disk space as work space. This is requested by specifying that the INPLACE sort intrinsic is to be used.

Logically, the CMS sort-merge software consists of the regular sort intrinsic, the inplace sort intrinsic, the merge intrinsic, and the sort language processor. The latter is a pre-pass to the intrinsics and is used when a stand-alone sort-merge is executed.

FUNCTIONAL DESCRIPTION

KEYS

A field within a record which is specified as containing a value to be used when the record is sorted or merged is a "key".

If several distinct fields within a record are specified then each field is a separate key, and the sort or merge required is a hierarchical multi-key sort or merge. The hierarchical ordering (that is, the relative order of importance) of the keys is determined by the order in which they are specified to the intrinsics.

In a multi-key sort or merge, each key may be specified to be ordered ascending or descending as required.

It is a CMS restriction that a key associated with a keyfile must be unsigned ascending and no greater than 28 bytes in length. It is further restricted to being a whole number of bytes in length and aligned on a byte boundary. These restrictions apply when the keyfile building function of the regular sort intrinsic is invoked (see KEYFILE CREATION).

With the exception of the latter type of invocation, the sum of the lengths of all keys (including signs) specified for use during any sort or merge must not exceed 29 bytes.

The maximum permissible number of keys for a sort or merge (except for the keyfile building function) is ten (10). A keyfile creation request must only specify one (1) key (a CMS keyfile may only relate to one key).

There are no restrictions on the contents of keys but note should be taken of the CMS deleted record format (see DELETED RECORDS) and of the CMS invalid index key values (see INVALID INDEX KEYS).

The CMS sort-merge intrinsics support the following key types:

- a) 4-bit unsigned numeric.
—each 4-bit unit is a binary coded decimal digit.
- b) 8-bit unsigned alphanumeric.
—an ordinary character field i.e. each byte contains an ASCII alphanumeric code.
- c) 4-bit signed numeric.
—each 4-bit unit is a binary coded decimal digit except that the most or least significant 4-bit unit is interpreted as a sign.
- d) 8-bit signed alphanumeric.
—each byte contains an ASCII alphanumeric code except that the most significant 4-bits of the most or least significant byte are interpreted as a sign.
- e) 8-bit alphanumeric with separate sign.
—each byte contains an ASCII alphanumeric code except that the most or least significant byte is interpreted as a sign.

Cases c) and d) assume the standard CMS sign conventions for 4-bit sign zones i.e. 0011 (#3) for positive and 0101 (#5) for negative; note however, that in practice a value other than #5 will be interpreted as positive.

Case e) assumes that the sign byte will contain the appropriate ASCII code for positive (#2B) or negative (#2D); note however, that in practice a value other than #2D will be interpreted as positive.

It should be noted that by definition cases b) and d) include 8-bit numeric (i.e. each byte contains the ASCII code for a decimal digit).

8-bit keys, with the exception of case e), may start on digit rather than byte boundaries.

A signed key may have the sign positioned at either end of the field but it must appear in the same position for all occurrences of the key.

Signed keys are ordered algebraically, that is, negative values are less than any positive value.

DELETED RECORDS

A deleted record in the CMS system is denoted by the fact that every byte in the record contains #FF. Most of the sort-merge functions will exclude either the deleted record itself or a reference to it from the sort-merge output file. A minority of the sort functions will not do this. The action taken by each sort-merge option is indicated at the relevant point in the following text.

REGULAR SORT INTRINSIC

Capabilities

The regular sort intrinsic may be invoked to perform one of the following four distinct functions.

Complete File Ordering

All the records contained within a designated input file will be ordered using a series of one or more specified keys.

Deleted records occurring in the input file will not be included in the output file.

The intrinsic will use disk work space of up to 2.2 times the size of the specified input file. In cases where the input file is resident on a user disk (cartridge or mini) the intrinsic may locate up to half the work space on that disk. In all other cases all work space will be resident on the system disk.

All sort disk work space will be returned to the system as free disk space prior to the sort going to (normal or abnormal) end of job.

Partial File Ordering

A partition (a contiguous group) of records contained within a designated input file will be ordered using a series of one or more specified keys.

Deleted records occurring in the input partition will not be included in the output files except in the case where the ordered partition is written back into the input file (overwriting the unordered copy of itself). The latter will occur only during a partition sort where the specified output file is the same as the specified input file, and the device type is disc. In such cases the deleted records will be suitably collated in the re-ordered partition.

The intrinsic will use disk work space of up to 2.2 times the input partition size. In cases where the file within which the input partition is located is resident upon a user disk (cartridge or mini) the intrinsic may locate up to half its work space on that disk. In all other cases all work space will be located on the system disk. All sort disk work space is returned to the system as free disk space prior to the sort going to (normal or abnormal) end of job.

Tagfile Creation

The intrinsic will create a tagfile corresponding to a designated input file using a series of one or more specified keys.

The intrinsic will create a file having the format of a CMS keyfile. However, unlike a keyfile, the tagfile will not contain any key values. The order in which the records will appear in the tagfile will correspond to the collating sequence of the data records to which they refer (based upon the key(s) specified for the sort). Deleted records occurring in the input data file will not be referenced in the tagfile.

Note that the data records contained in the specified input file will not be re-ordered by this sort option and that deleted records occurring in the file will not be removed.

The newly created tagfile will be given the same generation number as the specified input file.

The tagfile will be suitable for use as an ADDROUT file in RPG and for limited indexed access in COBOL (sequentially by the value(s) of the (original) key(s), that is, the tagfile is read sequentially). The legal indexed operations will be restricted to read and re-write.

The intrinsic will use disk work space of up to 2.2 times the size of the temporary file initially created by it. This file will contain records each of which will consist of the relative record number of the data record (in the input file) to which it refers and a copy of the data record's key value(s). In cases where the specified input file is located on a user disk (cartridge or mini) the intrinsic may locate up to half its work space on that disk. In all other cases all work space will be resident on the system disk. All sort disk work space will be returned to the system as free disk space prior to the sort going to (normal or abnormal) end of job.

Keyfile Creation

The intrinsic will create a keyfile corresponding to the designated input file using the specified unsigned key. Note that the specified key must total no more than 28 bytes in length, be a whole number of bytes, be aligned on a byte boundary and be specified to be collated ascending. The keyfile created will provide full CMS indexed access capability. Any deleted records occurring in the input data file will not be referenced in the keyfile (see also INVALID INDEX KEYS).

Note that the data records contained in the specified input file will not be re-ordered by this sort option nor will deleted records be removed.

The newly created keyfile will be given the same generation number as the specified input file.

Upon initiation of this sort option the acceptability or otherwise of duplicate key values in the keyfile may be specified. If they are unacceptable then the intrinsic will check for their presence. If a duplicate (or triplicate etc.) situation occurs the following action will be taken:

- a) the relative record number of each record concerned will be displayed on the SPO.
- b) the sort will continue but the output keyfile will be CLOSED with PURGE.

The intrinsic will use disk work space of up to 2.2 times the size of the temporary file initially created by it. This file will contain records each of which will consist of the relative record number of the data record (in the input file) to which it refers and a copy of the data record's key value. In cases where the specified input file is resident on a user disk (cartridge or mini) the intrinsic may locate up to half the work space on that disk. In all other cases all work space will be resident on the system disk. All sort disk work space will be returned to the system as free disk space prior to the sort going to (normal or abnormal) end of job.

Invalid Index Keys

It is a CMS restriction that the sequence of bytes in a record specified to be the key to be used during a keyfile creation sort may not:

a) consist of all binary zeros

or

b) contain any byte whose value is #FF.

If, during a keyfile creation sort, such a key is encountered the following action will be taken:

a) the relative record number of the record concerned will be displayed on the SPO.

b) the sort will continue but will create a keyfile that does not reference records containing an invalid index key.

Input Medium

The medium containing a specified input file to the regular sort intrinsic may be any one of the following:

a) 80—column cards.

b) Reel—to—reel tape.

c) Cassette.

d) Disk (cartridge or mini).

Input Restrictions

An input file must be wholly contained on one hardware type.

An input file must be of type data or of type source.

Upon initiation of the intrinsic the specified input file must have no write-permitted users.

Main Memory Requirements

For the purposes of speed and efficiency the intrinsic will make use of a non-overlayable memory area for work space. At initiation the size of this work area may be specified. The intrinsic will use the specified size unless it calculates that it is inadequate for a successful sort, in which case it will use the minimum acceptable size.

If the work space size is not specified to the intrinsic at initiation it will compute and use a size that it considers will produce a reasonable sort time while not unacceptably degrading overall system performance.

Output Medium

The medium specified to contain the required output file, with the exception of the keyfile and tagfile creation options, may be any one of the following:

a) 80—column cards.

b) Reel—to—reel tape.

c) Cassette.

d) Printer.

e) Disk (cartridge or mini).

Newly created keyfiles and tagfiles will always be made resident on a disk device.

With the exception of the keyfile and tagfile creation options, the type of the output file will correspond to the type of the input file (source or data).

Records that are output to a printer and that are smaller than a printer line will appear left justified with space fill on the right.

INPLACE SORT INTRINSIC

Capabilities

The inplace sort intrinsic will order the records contained within a designated input file on the key(s) specified. Note that the records are actually re-sequenced within the same file—a new output file is not created. The time taken will be substantially greater than that which the regular sort intrinsic would take to sort the same file.

Deleted records will not be removed; they will appear (appropriately located) in the re-ordered file.

Typically the inplace sort intrinsic will use disk work space of 0.2–0.3 times the size of the input file. All sort workspace will be returned to the system as free disk space prior to the sort going to (normal or abnormal) end of job.

Input Medium

The medium containing the specified input file to the inplace sort intrinsic must be disk (cartridge or mini).

Input Restrictions

An input file must be wholly contained on one disk.

An input file must be of type data or of type source.

Upon initiation of the intrinsic the input file must have a user count of zero.

Main Memory Requirements

The intrinsic will use a non-overlayable area of main memory for work space. The size of the area may not be specified at initiation.

Output Medium

The output file specified for an inplace sort must be the same file as was specified for input; therefore, like the input medium, the output medium must be disk (cartridge or mini).

MERGE INTRINSIC

Capabilities

The merge intrinsic will merge up to sixteen designated input files (using a series of one or more specified keys) and produce a designated output file.

Deleted records occurring in the input files will not be included in the output file.

The order in which the input files are specified to the intrinsic will determine the order in which records (from different input files) with the same key values will be ordered in the output file.

Disk Space Requirements

The merge intrinsic will not require any disk work space.

Input Medium

A file specified for input to the merge intrinsic may reside on any of the following media:

- a) 80—column cards.
- b) Reel—to—reel tape.
- c) Cassette.
- d) Disk (Cartridge or mini).

Input Restrictions

An input file must be wholly contained on one hardware type.

An input file must be of type data or source.

Upon initiation of the intrinsic no specified input file may have write-permitted users.

At any initiation of the merge intrinsic each physical non-disk hardware unit may contain not more than one specified input file. Any number of input files may be resident on the same disk.

The merge intrinsic will expect the record size associated with any particular input file to be equal to the record size associated with each of the other input files. It will also expect that the set of merge keys (which may consist of only one) will have the same position(s) and length(s) within a record and have the same format(s) for all the input files.

The merge intrinsic will assume that the records contained within each specified input file are already correctly ordered on the specified key(s). If the intrinsic detects that this is not the case, it will display an appropriate message on the system SPO detailing the input file in error and then go to abnormal end of job.

Main Memory Requirements

The merge intrinsic will use a non-overlayable area of main memory for work space. The size of this area may not be specified at initiation; it will always be approximately equal to the sum of the input files' block sizes plus twice the output file's block size.

Output Medium

The specified medium for the output file may be any one of the following:

- a) 80—column cards.
- b) Reel—to—reel tape.
- c) Cassette.
- d) Printer.
- e) Disk (cartridge or mini).

Records that are output to a printer and that are smaller than a printer line will appear left justified with space fill on the right.

The output file type will be made the same as that of the first specified input file.

INVOCATION OF THE SORT-MERGE INTRINSICS

GENERAL

The sort-merge intrinsics may be invoked by the following methods:

- a) by an executing S-program
- b) more directly by the user, by executing a stand-alone sort or merge.

Case a) will occur when sort or merge source constructs are used by the applications programmer.

Case b) requires the programmer to create a set of stand-alone sort language statements (see STAND-ALONE SORT-MERGE INITIATION) and to deliberately request the system (using the system control language) to execute SORT.

Regardless of the manner in which the sort-merge intrinsics are initiated they will always require certain information in a well defined format. This information will include the type of sort or order of merge, the sort or merge key(s), required disposition of the input file(s), special options, etc. This information will always be constructed by the CMS system on behalf of the user, and is not, therefore, defined in this manual.

COBOL GENERATED SORTS AND MERGES

A brief note of the actions invoked by use of the CMS COBOL sort and merge verbs follows; for more precise details the reader should consult the CMS COBOL manual.

Cobol Sort Verb

Use of this verb will always result in the initiation of the regular sort intrinsic for a full record sort on the whole of the specified file.

If the USING < file—name—2 > giving < file—name—3 > format is used then < file—name—2 > will be passed to the intrinsic as the input file and < file—name—3 > will be the newly ordered output file.

If the INPUT PROCEDURE, OUTPUT PROCEDURE option is used then an intermediate disk file will be created as records are RELEASED from the INPUT PROCEDURE. When control exits from the INPUT PROCEDURE the temporary file will be CLOSED with LOCK and passed to the intrinsic for sorting. After the sort has completed, the COBOL program will be restarted and as it executes RETURNS it will be passed records from a temporary disk file containing the original records correctly ordered.

Cobol Merge Verb

Use of this verb will always result in the initiation of the merge intrinsic to merge together the records contained within the specified input files to form a new file as indicated in the MERGE source statement.

STAND-ALONE SORT-MERGE INITIATION

A sort-merge intrinsic may be invoked as a free standing task either by an execution request from the system SPO or by a ZIP from an applications s-program. This is done by using the stand-alone sort language. The language allows the programmer to specify to the particular intrinsic invoked all relevant information concerning the job to be done. The language is processed by a dedicated module (the sort language processor) which will then invoke the required intrinsic.

Input Medium

The sort language processor will accept sort language statements from a card, cassette or disk (cartridge or mini) file. It will also accept the specification statements directly from an initiating message, that is, as a text string immediately following the system control command to execute the language processor. Whether initiated directly from the system SPO, or via a ZIP from an application task, the format of the text string must conform to the rules given in INVOKING THE SORT LANGUAGE PROCESSOR.

Details of the initiating message concept and the ZIP communicate may be found in the CMS Master Control Program (MCP) Reference Manual, form number 2007555.

Input Restrictions

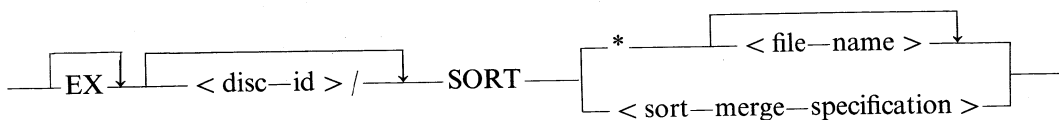
Any record size up to a maximum of 90 bytes can be chosen for the specification file (if present) when on a magnetic medium; for a card file the record size is defaulted to that of the physical record. If an initiating message is chosen as the specification method, its length is governed purely by the length of the specification statements (up to the CMS defined maximum of 255 characters). This restriction however does not apply to initiating messages generated by a ZIP from a user task to the sort language processor. The maximum length of such messages is implementation dependent.

As it may not be possible to completely specify a large merge in an initiating message of 255 characters, in such circumstances, the input to the processor should come from a file containing the merge statements.

Normally the processor will list the statements (together with any applicable error messages) on the printer; however, this may be inhibited by use of a particular sort language construct (see THE USER STATEMENT). The listing of the statements is automatically inhibited when the input comes from an initiating message.

The specification input file (if present) must be of type data or source.

Invoking the Sort Language Processor



The system control string used to initiate the sort language processor and hence a stand-alone sort or merge may optionally contain the name (preceded by an asterisk (*)) of the (card, cassette or disk) file containing the sort language statements or else (by omitting the asterisk) it may contain the sort language statements themselves. The <sort-merge-specifications> are described in the SORT LANGUAGE.

If the control string only contains an asterisk (but no following file name) the sort language processor will look for a file named SORTSPEC on the system disk.

If a file name is also present, then:

- 1) if it is a one part name containing seven characters or less a search of the following devices (in the specified order) will be executed for the named file:
 - a) Cards.
 - b) Cassette.
 - c) System disk.
- 2) if it is a two part name or one part containing more than seven characters a disk will be assumed and a check will be made for the presence of both the disk and the file.

If, in any of the above cases, the required file is not found the operator will be given the opportunity to load the file or DS the job.

THE SORT LANGUAGE

GENERAL

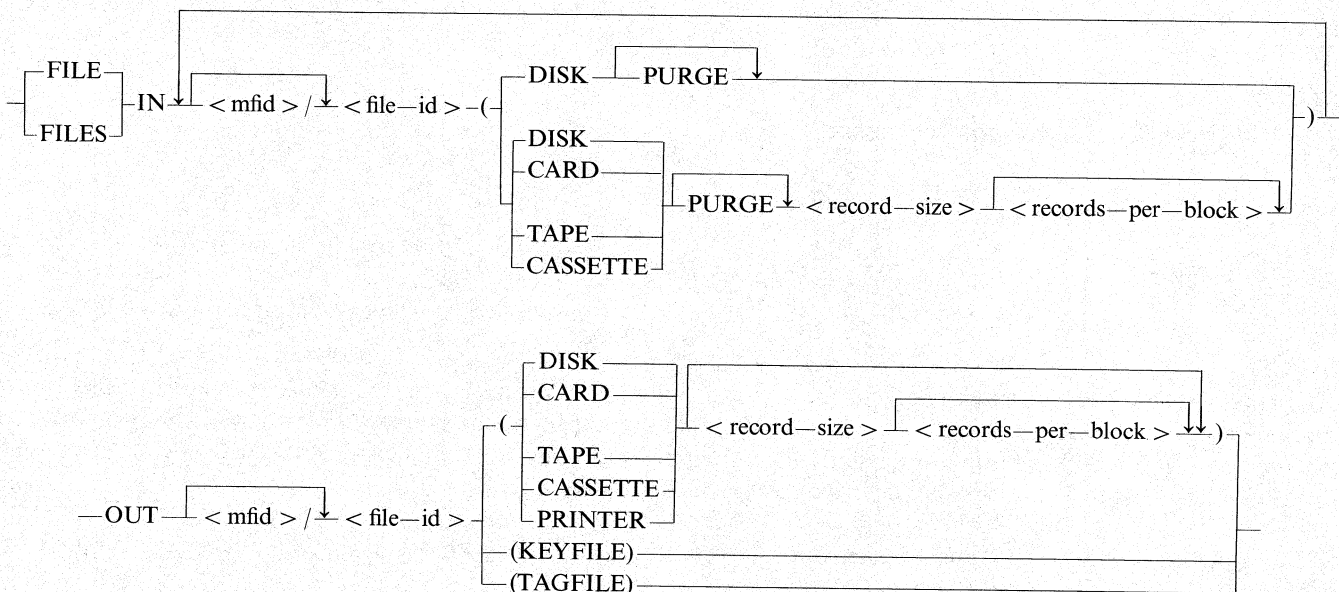
The sort language will allow the user to invoke any of the functions that the intrinsics provide. There are three sort language statements. These are:

- the File statement
- the Key statement
- the User Option statement

In any complete and valid group of sort language statements there will be one instance of the file statement, one instance of the key statement, and zero, one, or more clauses of the user option statement.

The statements are in free field format and may be arranged in any order consistent with the rules given below for each statement.

THE FILE STATEMENT



The file statement consists of two parts, the first of which describes the input file(s) and the second of which describes the output file. If more than one input file is specified then the MERGE clause of the user option statement must also be specified (see THE USER OPTION STATEMENT).

Each input file is described by a list of input descriptions. A sort request may only contain one input description; a merge request may contain up to sixteen. The parameters enclosed in parentheses must be separated from each other by at least one space.

The output file to be produced by sort-merge is specified by the output description. The parameters enclosed in parentheses must be separated from each other by at least one space.

Within a keyfile creation sort request:

- a) The output file < file—name > must refer to a disk file.
- b) The output parameters enclosed in parentheses must be replaced by the reserved word KEYFILE.

A newly created keyfile will always be made resident on a disk: the record size and blocking factor are not user definable.

Within a tagfile creation sort request:

- a) The output file < file—name > must refer to a disk file.
- b) The output parameters enclosed in parentheses must be replaced by the reserved word TAGFILE.

A newly created tagfile will always be made resident on a disk; the record size and blocking factor are not user definable.

The medium upon which a specified input file is resident is indicated by the appropriate reserved word, DISK, CARD etc. The required medium type of the output file is similarly specified. When the medium specified is DISK and there is no < mfid > present then the system disk is assumed.

The PURGE option is used to indicate that an input file is to be purged after use.

For both sort and merge input files, when the input medium is DISK, the < record size > and < records—per—block > parameters may optionally be omitted for each input file description. This applies to all options of the sort and merge functions. In the case of a merge, it is possible to intersperse disk input file descriptions which omit these parameters with some that do not.

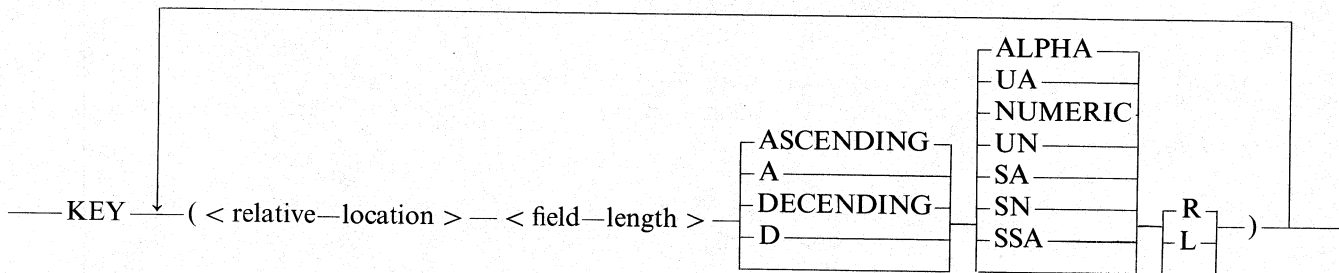
Irrespective of the output medium, for both record sort and merge output files, it is always possible to omit the < record size > and < records—per—block > parameters from the output file description. In the case of a record sort, the values assumed will be those possessed by the input file (regardless of whether they were also omitted from the input file description). For a merge, the record and block sizes of the first specified input file (with the same non specification proviso as for the sort) will be used if the output parameters are omitted.

In all cases (except an index sort) the input and output files must have the same record sizes.

The < records—per—block > entry may be omitted from both input and output file specifications in all cases where the < record size > is specified. In such conditions, its omission is taken to imply a blocking factor of one (1).

The < record—size > and < records—per—block > entries must be unsigned integers.

THE KEY STATEMENT



The key statement is used to define the key(s) that the sort-merge is to use to collate the input records.

The elements of each key description must be enclosed in parentheses and each must be separated from the next by at least one space.

A number of keys may be specified, each key description being enclosed in parentheses. The first key will be the major key and any additional keys will be minor keys of decreasing significance.

The < relative location > is used to specify the position of the key relative to the start of the record. The unit of measurement is a digit (that is, a 4-bit unit). The first 4-bit unit in the record is taken to be relative position one. The relative position of the left-hand 4-bit unit of the key (which, depending upon the key format, may contain a sign) is used to specify the key location.

The length of the key is indicated by the < field length >. The unit of measurement is the 4-bit unit. In the case of signed keys the length must include the sign.

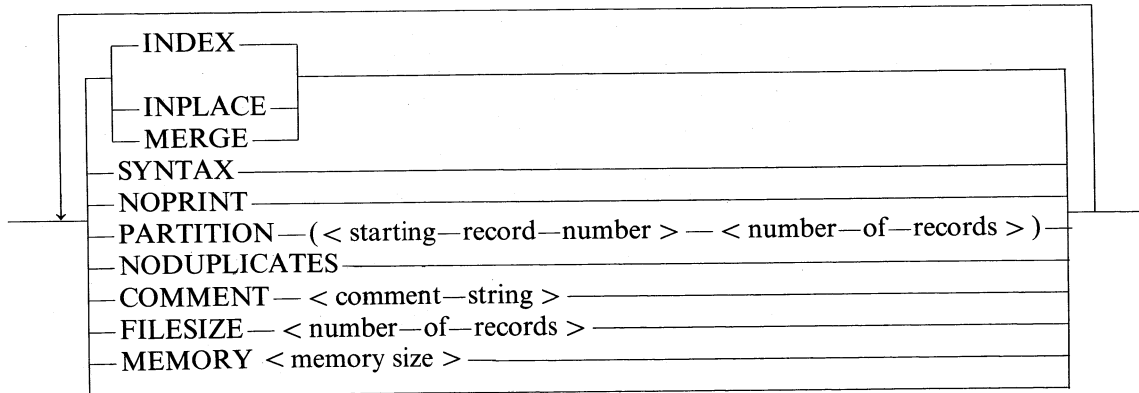
The sequence reserved words specifies whether the key is to be collated ascending or descending. Ascending sequence (ASCENDING or A) need not be specified as this is the default. DESCENDING or D may be used to indicate descending collation.

The format reserved words describe the format of the data within the key. ALPHA or UA indicates that the data is unsigned 8-bit alphanumeric. This need not be specified as it is the default. NUMERIC or UN indicate that the data is unsigned 4-bit numeric (that is, each 4-bit unit contains a binary coded decimal digit). SA specifies signed 8-bit alphanumeric. The sign is assumed to be resident in the most significant 4-bits of the most or least significant byte as given by the sign position character (see later). SN indicates that the data is signed 4-bit numeric. The sign is assumed to be resident in the most or least significant 4-bits as given by the sign position character. SSA indicates that the data is 8-bit alphanumeric with a leading or trailing 8-bit separate sign (the sign position will be indicated by the sign position character).

The sign position character is used in the description of a signed key to specify whether the sign is located at the left-hand (most significant) end or the right-hand (least significant) end of the key. If the sign position character is absent within the definition of a signed key then the left-hand end is assumed.

For a full discussion of sort-merge key types and sign zone interpretation refer to KEYS.

THE USER OPTION STATEMENT



The purpose of the user option statement is three fold:

- a) to specify to the intrinsics (where necessary) which particular function is required.
- b) to tailor a sort or merge to the particular target machine configuration (amount of main memory, printer availability, etc.).
- c) to add comments to the sort language statements.

A valid set of sort language statements need not contain any part of the user option statement.

The clauses of the user option statement may be coded in any order relative to each other and to the file and key statements.

The sort type option is used to indicate which sort-merge function is required. If the clause is omitted then a regular full record sort is assumed. The INDEX sort type option will cause either a keyfile or a tagfile to be created for the specified input file. The choice will depend upon the value of the output file parameters in the file statement (see THE FILE STATEMENT). The INPLACE sort type option causes the execution of a full record sort using a minimal amount of disk work space. The MERGE sort type option causes the contents of the specified input files to be merged together producing the required output file.

By use of the SYNTAX option, the programmer may specify that he only requires his sort language statements to be checked for correctness and does not wish an intrinsic invoked.

The NOPRINT option may be used to inhibit the listing of the sort language statements on the printer. NOPRINT should be the first entry in the specification statement list. If the statements are input via an initiating message (see INPUT MEDIUM), they are treated as if the NOPRINT option is set, that is, the listing is always inhibited. If the sort language processor detects an error in the statements while the NOPRINT option is set, the error number (as defined in SORT LANGUAGE PROCESSOR MESSAGES) will be printed on the system SPO. Note that warning message numbers (where corrective action is taken) will not appear on the SPO. A maximum of five error numbers will be displayed.

The PARTITION option is used to specify that only a particular part of the input file is to be sorted. The < starting record number > and < number of records > must be enclosed in parentheses and separated by at least one space. A partition is always inclusive of the < starting record number >. If < number of records > is zero then all records between < starting record number > and end-of-file are sorted. The PARTITION option may not be used with the INDEX or MERGE options.

Within a keyfile creation request, the NODUPLICATE option may be used to specify whether or not duplicate records are allowable in the required keyfile. Absence of the clause will imply that they are. This option is not valid in other than a keyfile creation request.

The COMMENT option is used to document a sort language statement list. A comment is delimited by the reserved word COMMENT and the end of a record (or the end of the message if an initiating message is used). Comments may appear between statements and between clauses of the user option statement. They may also appear between file

descriptions in the file statement and key descriptions in the key statement provided that the syntactic rules of these statements are not contravened.

An estimate of the size of the requested output file may be specified to the intrinsic by use of the FILESIZE option. The value given should be approximately equal to (or greater than) the number of sort-merge records that the intrinsic will produce. This will aid disk space optimisation whilst a larger value than the true output file size will enable the file to be subsequently extended by the addition of extra records. Where possible (that is, input files on disk) the intrinsic will calculate a minimum size for the specified output file and use this value in preference to the FILESIZE option, if the calculated value is larger than the option size. Any inaccuracy in the option is therefore not fatal to the sort-merge. If the FILESIZE option is not given, the intrinsic will either assume a default file size (all input files not resident on disk) or calculate an optimum value (all input files resident on disk). The presence of this option is not a prerequisite for a successful sort-merge. This option is not applicable to the INDEX or PARTITION (when the sorted section is inserted back into the original file) functions of the regular intrinsic or to the INPLACE intrinsic.

The programmer may specify to the regular sort intrinsic how much main memory it may use as a non-overlayable work area by use of the MEMORY option. If the intrinsic calculates that the amount specified is less than the minimum required for a successful sort it will ignore the programmer's directive and use the minimum satisfactory size.

SORT LANGUAGE PROCESSOR MESSAGES

GENERAL

Sort language processor messages are divided into two classes, viz. ERROR messages and WARNING messages. ERROR messages are further subdivided into two sections, that is, SYNTAX errors where the grammar of the language has been contravened and SEMANTIC errors where an inconsistency has been detected in the parameters specified (for example, key position outside the record boundary). In the case of WARNING messages, corrective action is always attempted. The action taken is explained in the appropriate message.

The messages are numbered from 0 to 99. Those in the range 0 to 39 are warnings, whilst those from 40 onwards fall into the two error classes, that is, 40 to 59 are syntax messages and 60 to 99 are semantic messages.

Messages appearing below that are followed by a series of dots (...) should be read with the phrase NEAR COL XXX inserted in place of the dots. All messages are sent to the printer.

WARNING MESSAGES

Number Message

0	EXPECTED SLASH NOT FOUND, "/" INSERTED ...
1	EXTRA "FILE IN" ...
2	INPUT, OUTPUT FILES IDENTICAL, < PURGE OPT > INSERTED
3	OVERLENGTH PART OF < LABEL NAME > IGNORED ...
4	IGNORE < FILE SIZE OPT > SINCE LESS THAN < PARTIT OPT >
5	EXPECTED BRACKET NOT FOUND, "(" INSERTED ...
6	< DUPLICATE OPTION > VALID IN INDEX-KEYFILE SORT ONLY
7	EXPECTED BRACKET NOT FOUND, ")" INSERTED ...
8	ILLEGAL TO DELETE INPUT FILE, < PURGE OPT > IGNORED
9	PARTITN TO IN-FILE, ALTERED OUT-FILE PARAMS IGNORED
10	< USER OPTION > ALREADY INVOKED, LATEST USE ...
11	MERGE < SORT TYPE OPTION > NOT SPECIFIED
12	< PARTITION OPT > VALID FOR INPLACE/REGULAR SORT ONLY
13	MISSING "FILE IN" ...
14	INDEX < SORT TYPE OPTION > NOT SPECIFIED
15	EXTRA "KEY" ...

16 MERGE INTRINSIC IGNORES < FILE SIZE OPTION >
 17 MISSING "KEY" ...
 18 INPLACE INTRINSIC IGNORES < MEMORY OPTION >
 19 < M—FILE/DP ID > IGNORED ON NON—MAGNETIC MEDIA FILE ...
 20 NUMBER TOO BIG, MAXIMUM VALUE ALLOWABLE ASSUMED ...

21 NOT NECESSARY TO PURGE CARD FILE ...
 22 < SIGN POSITION > GIVEN FOR UNSIGNED KEY ...
 23 FIRST UNIT NUMBERED 0 RATHER THAN 1 ...

24 < FILE SIZE OPT > IGNORED SINCE OUT OF RANGE ...
 25 MERGE INTRINSIC IGNORES < MEMORY OPTION >

26 < BLOCK FACTOR > OF 0 NOT ALLOWED, 1 ASSUMED ...
 27 IN— & OUT—FILE RECORD SIZES MADE EQUAL
 28 < BLOCK FACTOR > TOO LARGE, MAXIMUM ASSUMED ...
 29 INPLACE SORT MUST HAVE IDENTICAL IN— & OUT—FILES
 30 PARTITION SIZE TOO BIG, SORT TO EOF ASSUMED ...

ERROR MESSAGES

Number Message

40 < KEY STATEMENT > ALREADY PROCESSED, NOW ...
 41 < DIGIT STRING > EXPECTED ...
 42 < CHARACTER STRING > EXPECTED ...
 43 < SEPARATOR STRING > EXPECTED ...
 50 NO < FILE STATEMENT > SPECIFIED
 51 STATEMENT PARTLY IGNORED, ILLEGAL WORD ...
 52 < LETTER STRING > EXPECTED ...
 53 MISSING "< .. > / < .. >" OR "< .. >" ...
 54 UNSUPPORTED < IN/OUT MEDIA > ...
 55 UNSUPPORTED < SORT TYPE OPTION > ...
 56 PART OF < FILE STATEMENT > MISSING, NOW ...
 57 NO < KEY STATEMENT > SPECIFIED
 58 < FILE STATEMENT > ALREADY PROCESSED, NOW ...
 59 REMAINDER OF STATEMENT MISSING

Number Message

60 TOO MANY KEY SPECIFICATIONS ...
 61 TOO MANY FILE SPECIFICATIONS ...
 62 INPUT FILES RECORD SIZES NOT IDENTICAL ...
 63 < RECORD SIZE > OUT OF RANGE ...
 64 EXTRA DIGITS IN OVERLENGTH STRING IGNORED ...
 65 (CURRENT SUM OF) KEY LENGTH(S) OUT OF RANGE ...
 66 MIN LENGTH OF SN KEY IS TWO 4-BIT UNITS ...
 67 BUFFER SIZE TOO LARGE ...
 68 DUPLICATE < IN—FILE PARAMS >, LATEST INSTANCE ...
 69 BUFFER SIZE TOO BIG FOR < IN/OUT MEDIA > ...
 70 ONLY ONE IN—FILE LEGAL FROM MULTIFILE TAPE ...
 71 MERGE INTRINSIC NEEDS AT LEAST 2 INPUT FILES

72 INDEX PARAM MUST BE "OUT...(KEYFILE/TAGFILE)"
 73 KEY OVER—RUNS RECORD BOUNDARY
 74 ILLEGAL TO OVERWRITE INPUT FILE WITH TAG/KEY FILE
 75 ALPHANUMERIC KEY LENGTH NOT EVEN NUMBER OF 4-BITS...
 76 < MEDIA > MUST BE DISK IF SORTING BACK TO IN—FILE

77 IN— & OUT—FILE RECORD SIZES MUST BE IDENTICAL
 78 INDEX—KEYFILE KEY LENGTH NOT EVEN NUMBER OF 4—BITS
 79 ONLY ONE KEY LEGAL IN INDEX—KEYFILE SORT
 80 INDEX—KEYFILE SORT KEY TOO LONG
 81 INDEX—KEYFILE SORT KEY MUST BE "... A UA/UN)"
 82 ONLY INDEX SORT CAN SPECIFY "KEYFILE/TAGFILE"
 83 INDEX—KEYFILE SORT KEY MUST START ON BYTE BOUNDARY
 84 MIN LENGTH OF SSA KEY IS FOUR 4—BIT UNITS...
 85 SSA KEY MUST START ON BYTE BOUNDARY...

SORT—MERGE INTRINSIC MESSAGES

GENERAL

The intrinsic messages are a mixture of error and information messages. Each message is preceded by the phrase < mix number > / < program name > and they are all directed to the system SPO. Note that individual message numbers are not printed.

INTRINSIC MESSAGES

Number Message

170 DUPLICATE RECORD < record number >
 171 ILLEGAL INDEX KEY IN RECORD < record number >
 172 RECORDS LOST OR GAINED BY SORT-MERGE
 173 < number > DUPLICATE RECORDS
 174 < number > RECORDS CONTAINING INVALID INDEX KEYS
 175 < number > RECORDS DELETED
 176 < number > RECORDS MERGED
 177 < number > FILES MERGED
 178 SORT-MERGE OUTPUT FILE NOT CREATED
 179 SORT-MERGE ABNORMAL EOJ
 180 SORT-MERGE SOFTWARE ERROR
 181 < number > RECORDS REFERENCED BY KEYFILE/TAGFILE
 182 NO INITIATING MESSAGE
 183 < number > RECORDS SORTED
 184 FILE ERROR (< number >) ON SORT-MERGE FILE < file name >
 185 UNORDERED MERGE INPUT FILE < file name >
 186 TOO MANY RECORDS FOR SORT-MERGE
 187 DUPLICATE RECORDS—KEYFILE NOT BUILT
 188 INIT MESSAGE INVALID
 189 SORT-MERGE INITIATED FROM < mix number > / < program name >

Message 184 represents differing file errors depending upon the value of < number >. Defined meanings are as follows:

1. EOF on output file
2. parity on input file
3. EOF on sort workfile
4. bad disk address
5. sort workfile error
6. input file error
7. output file error.

SORT LANGUAGE RESERVED WORDS

/	DECENDING	KEYFILE	PURGE
(DISC	L	R
)	DISK	MEMORY	SA
A	FILE	MERGE	SN
ALPHA	FILES	NODUPLICATES	SSA
ASCENDING	FILESIZE	NOPRINT	SYNTAX
CARD	IN	NUMERIC	TAGFILE
CASSETTE	INDEX	OUT	TAPE
COMMENT	INPLACE	PARTITION	UA
D	KEY	PRINTER	UN



SECTION 5

B80 DEPENDENT ROUTINES

FUNCTIONAL SUMMARY

The B80 processor executes machine language instructions (micro-instructions) which are taken by the processor from main memory. The B80 main memory consists of 4K bytes of Read-only Memory (ROM) and up to 60K bytes of Random Access Memory (RAM). ROM will retain its contents when power is removed from the system, whereas RAM will not.

When power is applied to the system, or when the Load Enable button is pressed, the contents of RAM are deemed invalid. Execution control is passed to the permanent routines in ROM. The main function of the ROM routines is to load RAM with, and pass control to, executable system software.

The ROM routines provide the ability to load system software from cassette or from disc, selected by Program Key (PK) depression. PK1 will cause entry to the cassette load permanent ROM routine. This routine is provided for ACSYS use, and is described in the B80 ACSYS System Software Operation Guide, form number 2007639. PK2 will cause entry to the disc load permanent ROM routine, which will search for the B80 Bootstrap on a correctly formatted B80 CMS disc. The facilities provided by the B80 Bootstrap are described below.

CMS B80 BOOTSTRAP

The CMS B80 Bootstrap occupies sectors 2 through 9 on all B80 initialised CMS discs. The bootstrap provides the following facilities, which may be invoked by the indicated PK.

- PK3 Warmstart MCP (See Warmstart Procedure)
- PK4 Memory Dump to cassette (See Memory Dump)
- PK5 Memory Dump to disc (See Memory Dump)
- PK6 Load Stand-Alone Utility from disc (See Stand-Alone Utility)

Bootstrap Load

The following procedure will cause the B80 Bootstrap to be loaded and executed.

- 1) Apply power to the system or press the Load Enable button if power is already applied to the system. The power on switch and the Load Enable button are located behind the processor door. The power on switch is below and to the left of the lower switch panel, and should be transferred upwards to apply power to the system. The Load Enable button is the "bell push" type switch on the right of the top panel immediately behind the processor door. The system will illuminate all keyboard indicators, then sequentially, with decreasing time interval, extinguish them, leaving the lights associated with PK1 and PK2 illuminated. PK1 will cause entry to the Cassette Load permanent ROM routine and PK2 will cause entry to the Disc Load permanent ROM routine
- 2) Install a WRITE-ENABLED B80 initialised CMS disc in any suitable disc drive and make the drive ready. The B80 Equipment Reference Manual, form number 2007233 gives full details of the insertion/reading procedure for each type of disc drive which may be used with the system.
- 3) Press PK2. PK2 is the rightmost blue key below the *illuminated* keyboard indicators, that is, the second key from the left in the row of blue keys. The Disc Load permanent ROM routine will search the available disc drives for a B80 Bootstrap routine. The search will commence with the bottom disc drive of the disc drive unit with the highest channel address. If no bootstrap is found on that disc, the search will continue with the top drive of that channel, then the bottom drive of the next lower channel address, etc., until a bootstrap is found. If no bootstrap is found, an error pattern is displayed on the keyboard indicators as described in POSSIBLE ERRORS. The channel location of each peripheral device may be found by consulting the initial serial printer output produced by the CUSTOMER CONFIDENCE ROUTINE. When a bootstrap is found, it is loaded into Random Access Memory (RAM), and execution control is passed to it. The bootstrap will illuminate the keyboard indicators associated with PK's 3 through 6. These PK's will invoke the following functions which are described in detail in the indicated subsection of this section.

- PK3 Warmstart MCP (WARMSTART PROCEDURE)
- PK4 Memory Dump to Cassette (MEMORY DUMP)
- PK5 Memory Dump to Disc (MEMORY DUMP)
- PK6 Load Stand-Alone Utility (STAND-ALONE UTILITY)

Possible Errors

The following errors may be encountered whilst loading the bootstrap.

- 1) The sequential extinguishing of keyboard indicators does not occur.
 - The MTR switch (behind the processor door, above and to the right of the power on switch) is in the MTR position. Transfer the MTR switch to NORMAL and press Load Enable.
- 2) PK2 is ignored.
 - The keyboard is locked in "shift" mode. Press the shift key below the shift-lock key, and press PK2 again.

- 3) The numeric light illuminates instead of PK3 through PK6.
—PK1 was pressed. Press Load Enable then PK2.
- 4) The keyboard error light illuminates, and a keyboard light pattern is displayed when PK2 is pressed.
—If the lights D1 through D8 are illuminated, no bootstrap was found. Check that a correct disc is installed and ready, press Load Enable, and select PK2 again.
—If one of the lights D1 through D8 is extinguished, a disc error was encountered whilst attempting to locate a bootstrap. The *extinguished* D light indicates the channel number of the disc in error. D1 corresponds to channel 0, D8 corresponds to channel 7. Other keyboard lights indicates relevant information as shown in the table below. Take a note of the light pattern for field engineering use. Power off the faulty disc and replace with a back up copy. Press Load Enable, then PK2.

Note: Error 4 above may be caused by a disc drive fault. This may be checked by using the disc in another drive. If the bootstrap successfully loads, run the Customer Confidence Routine (see CUSTOMER CONFIDENCE ROUTINE) and notify field engineering.

Table of Keyboard Indicators

Channel Indicators

D1—Channel 0	D5—Channel 4
D2—Channel 1	D6—Channel 5
D3—Channel 2	D7—Channel 6
D4—Channel 3	D8—Channel 7

If an error is encountered whilst the ROM Disc Load Routine is attempting to locate bootstrap, the D light corresponding to the channel in error will be extinguished. All other D lights will be illuminated. The PK lights have the significance shown below. If all D lights are illuminated, no bootstrap was found, and the PK lights have no significance.

Primary Status Indicators

PK1—Off indicates top drive; On indicates bottom drive
 PK2—Off indicates SEEK COMPLETE
 PK3—Off indicates END OF CYLINDER
 PK4—Off indicates SEARCH COMPLETE
 PK5—Off indicates SECONDARY STATUS Condition (below)
 PK6—On indicates OPERATIONAL
 PK7—Off indicates SEEK INCOMPLETE
 PK8—On indicates GOOD STATUS (will not be seen).

Secondary Status Indicators

PK9 —On indicates EQUAL
 PK10—On indicates ON CYLINDER
 PK11—Off indicates ILLEGAL SEEK
 PK12—On indicates WRITE INHIBIT
 PK13—Off indicates SECTOR NOT FOUND
 PK14—Off indicates LRC ERROR (Parity)
 PK15—Off indicates ILLEGAL COMMAND SEQUENCE
 PK16—Off indicates DEVICE ERROR.

These indicators are only significant if SECONDARY STATUS (PK5 above) is OFF.

Retry Count Indicators

The indicators associated with PK17 through PK24 indicate the number of retries attempted before an error is declared.

STAND-ALONE UTILITY

The B80 CMS Stand-Alone Utility is micro-programmed and supplied on disc as a single file. It is not portable to other CMS machines. The file contains a number of functions, any of which can be used without reloading the utility. The functions are: initialise a disc to CMS format, reformat a disc, load a disc from cassette, enter Warm Start, copy files from disc to disc, delete disc files, list disc file names, list cassette names, and initialise an MTR disc. Each function is requested by typing a command message terminated by an OCK. The Ready Request key is not required prior to keyboard input.

LOADING THE UTILITY

The following procedure must be followed to load the Stand-Alone Utility.

- 1) Perform the bootstrap load procedure using a B80 CMS disc containing the Stand-Alone Utility, file identity "SAU". The bootstrap load is fully described in **BOOTSTRAP LOAD**, but the salient features are:
 - Apply power to the system (and disc drives)
 - Install the disc and ready the unit
 - Press PK2

The system should halt with the keyboard indicators associated with PK's 3 through 6 illuminated.

- 2) Press PK6. PK6 is the sixth key from the left in the row of blue keys. The bootstrap routine will search the disc directory of the disc from which the bootstrap was loaded for a system file called "SAU". If no such file is found, and another disc containing bootstrap is located at a lower unit address then the ROM disk loader will be invoked automatically, and this step should be repeated when PK's 3 through 6 are illuminated. If no disc on the system contains the file "SAU", then an error indicator as described in Possible Errors, below, is displayed. When the Stand-Alone Utility is located, it is loaded to RAM and execution control is passed to it. Completion of a successful load is indicated by the printing of the start up message.

Possible Errors

The following errors may be encountered whilst loading the Stand-Alone Utility.

- 1) The sequential extinguishing of keyboard indicators does not occur.
 - The MTR switch (behind the processor door, above and to the right of the power on switch) is in the MTR position. Transfer the MTR switch to NORMAL, and press Load Enable.
- 2) PK2 is ignored.
 - The keyboard is locked in "shift.. mode. Press the shift key below the shift-lock key, and press PK2 again.
- 3) The numeric light illuminates instead of PK's 3 through 6.
 - PK1 was pressed. Press Load Enable then PK2.
- 4) The keyboard error light illuminates, and a keyboard light pattern is displayed when PK2 is pressed.
 - If the lights D1 through D8 are illuminated, no bootstrap was found. Check that a correct disc is installed and ready, press Load Enable, and select PK2 again.
 - If one of the lights D1 through D8 is extinguished, a disc error was encountered whilst attempting to locate a bootstrap. The *extinguished* D light indicates the channel number of the disc in error. D1 corresponds to channel 0, D8 corresponds to channel 7. Other keyboard lights indicate relevant information as shown in the table below. Take a note of the light pattern for field engineering use. Power off the faulty disc and replace with a backup. Press Load Enable, then PK2.

- 5) The keyboard error light illuminates, and a keyboard light pattern is displayed when PK6 is pressed.
- Bootstrap has been loaded from a disc, but SAU could not be loaded from that disc. This may be caused if a disc error exists, or if no disc on the system contains an SAU file. The D light which is *illuminated* indicates the channel number of the disc in error. D1 corresponds to channel 0, D8 corresponds to channel 7. Other keyboard lights indicate relevant information as shown in the table below. Note that both a bootstrap and an SAU must be present on the same disc. If a suitable disc is present on the system, take a note of the light pattern for field engineering use. Power off the disc and replace with a backup copy. Press Load Enable, PK2, and PK6.

Note: Errors 4 and 5 above may be caused by a disc drive fault. This may be checked by using the disc in another drive. If the disc successfully loads, run the Customer Confidence Routine (see CUSTOMER CONFIDENCE ROUTINE) and notify field engineering.

- 6) All keyboard indicators alternatively illuminate and extinguish.
- PK4 or PK5 was pressed during the load, and the system has just been powered up. Random Access Memory contains parity errors which have not been deleted by execution of MCP. Press Load Enable, PK2, and PK6.
- 7) The load stops with all keyboard indicators illuminated.
- The serial printer is not ready. Check that the forms transport is properly closed, and the printer cover is properly latched. The variable forms spacing lever (60 cps printer) may be unlatched. The utility will continue when the fault has been corrected.

Table of Keyboard Indicators

Channel Indicators

D1—Channel 0	D5—Channel 4
D2—Channel 1	D6—Channel 5
D3—Channel 2	D7—Channel 6
D4—Channel 3	D8—Channel 7

If an error is encountered whilst the ROM Disc Load Routine is attempting to locate bootstrap, the D light corresponding to the channel in error will be extinguished. All other D lights will be illuminated. The PK lights have the significance shown below. If all D lights are illuminated, no bootstrap was found, and the PK lights have no significance.

If an error is encountered whilst the bootstrap is attempting to locate SAU, the D light corresponding to the channel in error will be illuminated. All other D lights will be extinguished. The PK lights have the significance shown below.

Primary Status Indicators

- PK1—Off indicates top drive; On indicates bottom drive
- PK2—Off indicates SEEK COMPLETE
- PK3—Off indicates END OF CYLINDER
- PK4—Off indicates SEARCH COMPLETE
- PK5—Off indicates SECONDARY STATUS Condition (below)
- PK6—On indicates OPERATIONAL
- PK7—Off indicates SEEK INCOMPLETE
- PK8—On indicates GOOD STATUS (will not be seen).

Secondary Status Indicators

PK9 —On indicates EQUAL
PK10—On indicates ON CYLINDER
PK11—Off indicates ILLEGAL SEEK
PK12—On indicates WRITE INHIBIT
PK13—Off indicates SECTOR NOT FOUND
PK14—Off indicates LRC ERROR (Parity)
PK15—Off indicates ILLEGAL COMMAND SEQUENCE
PK16—Off indicates DEVICE ERROR.

These indicators are only significant if SECONDARY STATUS (PK5 above) is OFF.

Retry Count Indicators

The indicators associated with PK17 through PK24 indicate the number of retries attempted before an error is declared.

OPERATION

The Stand-Alone Utility is loaded as described above. Completion of a successful load is indicated by the message:

```
STAND-ALONE UTILITY
VERSION < version—number >
REQUEST "HELP" FOR FUNCTION SUMMARY
```

The utility then enters the Function Select State. Any Stand-Alone function may be selected by typing the appropriate command as detailed in FUNCTION DESCRIPTIONS, and terminating with an OCK. At the completion of the function (except warm-start, WS) control returns to the Function Select State with the prompt "FUNCTION"? The utility is terminated either by entering warm-start through the WS command, or by pressing the Load Enable button.

Use of the backspace key (shifted or unshifted) during keyboard input will backspace the print head and delete the last character entered.

Use of the Reset key during keyboard input allows invalid input to be cancelled. The current message is ignored, and the console forms are advanced one line.

Invalid Input

Invalid input from the keyboard will either be ignored with an appropriate message, for example:

```
INVALID REQUEST
or a prompt to the operator will be repeated.
```

Note: The drive containing the disc to be initialised is identified by:

DFA, DFB, DFC Burrough's Super Mini-Disc
DKA, DKB, DKC Cartridge Discs.

FUNCTION DESCRIPTIONS

The following functions are provided in the Stand-Alone Utility. Once a function has been initiated, it must be taken to completion before another function may be executed (that is, no multi-tasking is permitted).

- 4) **DUPLICATE FILE NAME — < file—id > .**
This message is displayed if a file of the displayed name already exists in the directory of the destination disc. The function will continue if there are further files to be copied.
- 5) **NAME LIST FULL.**
This message is displayed if there is no room in the directory of the destination disc for the name of the file. The function ENDS.
- 6) **O/P ERROR— < file—id > .**
This message is displayed if a write error is encountered on the destination disc during a copy. The function will continue if there are further files to be copied.
- 7) **I/P ERROR— < file—id > .**
This message is displayed if a read error is encountered on the source disc during a copy. The function will continue if there are further files to be copied.
- 8) **CANNOT ALLOCATE AREAS FOR— < file—id > .**
This message is displayed if there is no available area of appropriate size on the destination disc for the specified file. The function will continue if there are further files to be copied.
- 9) **Dxy DIRECTORY I/O ERROR.**
This message is displayed if a read or write error is encountered whilst attempting to access the directory of the specified disc. The directory structure of an input disc may be corrupted.
x identifies the type of disc: F—Burroughs Super Mini-disc; K—Cartridge Disc.
y identifies the unit: A, B, C, etc.
The function ABORTS.
- 10) **O/P DISC IS NOT WRITE PERMIT.**
This message is displayed if the destination disc is write inhibited. The write lockout hole on Burroughs Super Mini-Disc should be covered. The write lockout plug on cartridge disc should be flush with the outer surface of the cartridge.
The function ENDS.
- 11) **Dxy DEVICE ERROR.**
This message is displayed if a hardware error is encountered during copy. Dxy is as specified in 8 above. The message will be followed by one of the following displays:
DRIVE INOPERABLE
OFF CYLINDER
SEEK TIMEOUT
CONTROLLER PROBLEM.

These messages indicate a malfunction in the specified drive unit. The customer confidence routine (see CUSTOMER CONFIDENCE ROUTINE) should be executed, and field engineering should be notified. Note that DEVICE INOPERABLE will be displayed if the specified drive is accidentally made not ready.

FE (Initialise MTR Disc)

—FE—

A virgin or a formatted disk is initialised to CMS format with suitable sectors reserved for MTR test routines. (These sectors will be denoted as BAD sectors on a KA map of the disk.) The surface is checked by writing and reading test patterns to each sector. Bad sectors and the six MTR tracks are made unavailable. A disk label is written and the file directory is created with a single, SYSMEM, entry. Sectors 1 through 31 are loaded from a file "CMSBOOTxxxx" which can be contained by any on-line disk. The correct identification string must be found in sector one before loading can begin.

The xxxxx characters are ignored by the utility; only the seven leading characters are compared when the on-line discs are searched (that is, the file specification searched for is equivalent to CMSBOOT=). The discs are searched in descending channel order, cartridge discs first, then a further scan of Burroughs Super Mini Discs.

PROMPTS AND RESPONSES

The function will request the necessary input by means of prompt messages as detailed below. Invalid input will cause the prompt to be repeated.

<u>PROMPT</u>	<u>INPUT FORMAT</u>	<u>REMARKS</u>
DRIVE	3 characters	The drive unit containing the disc to be initialised DFA, DKA, etc.
<p>Note: At this point the function will perform the disc surface test. The sector address of each track is displayed in binary upon the keyboard indicators (PK24 = 1, PK23 = 2, PK22 = 4 etc.). The disc surface test will take several minutes, the exact time depending upon the type of disc being initialised.</p>		
DATE	MM/DD/YY	Eight characters, for example 01/01/76
FILES	Up to 3 numerals less than 255	The maximum number of files which the disc is to contain. This number determines the number of sectors to be allocated as the disc directory.
SERIAL	6 numerals	The user's identifying serial number.
PACK	Up to 7 characters	The name by which the disc will be known to the system, that is, the < disc—id >.
OWNER	Up to 14 characters	Any identifying character string.
RESTRICTED	Y or N	This option is included for future CMS implementations. The normal response is N.

OUTPUT MESSAGES

The following messages may be output by the utility during the execution of the function.

Normal Execution

The message

NO. OF BAD SECTORS: < integer >
is displayed when a disc has been successfully initialised.

Normal Termination

The message

END FE

is displayed, and the Function Select state is entered, whenever the function is discontinued. This will be described as "the function ENDS".

Errors During Execution

The following messages are displayed in the event of the corresponding error.

1) TRACK 0 BAD.

This message is displayed if any sector in Track 0 cannot be used. A CMS disc contains reserved information in Track 0 (for example, the warmstart bootstrap). The function will END.

2) Dxy DEVICE ERROR.

This message is displayed if a hardware error is encountered during FE.

x identifies the type of disc: F—Burroughs Super Mini-disc; K—Cartridge Disc.

y identifies the unit: A, B, C, etc.

This message is displayed if a hardware error is encountered during FE.

DRIVE INOPERABLE

WRITE INHIBITED

OFF CYLINDER

SEEK TIMEOUT

CONTROLLER PROBLEM.

DRIVE INOPERABLE may be caused if the associated disc is made not ready accidentally.

WRITE INHIBITED indicates that the disc in the specified drive has its write lockout indicator set. Ensure that the write lockout hole (Burroughs Super Mini-disc) is covered, or the write lockout plug (Cartridge Disc) is flush with the surface of the disc cartridge.

All other messages indicate a malfunction of the drive unit. The customer confidence routine (see CUSTOMER CONFIDENCE ROUTINE) should be executed, and field engineering should be notified.

3) LOAD CMS BOOTSTRAP FILE AND HIT OCK 1 TO CONTINUE.

This message is displayed if no file in the group CMSBOOT= can be found on any disc on the system. A suitable file may be LD'ed to any on-line disc, of a disc, or a disc containing a suitable file may be installed in an available drive.

4) BOOTSTRAP VERSION <bootstrap version string> USED

This message is displayed when the bootstrap file has been successfully written.

IN (Initialise Disc)

———— IN ————

It is a requirement of the Master Control Program that any disc to be used on the system must have a valid CMS disc label, and a CMS disc directory. In addition, each sector of the disc must be initialised with its address. The IN function will check the recording surface of the disc to be initialised by writing and reading test patterns in each sector of the disc. Bad sectors are made unavailable. The number of bad sectors detected is displayed to allow badly worn discs to be discarded. The function will write a disc label containing the parameters supplied by the responses to the appropriate prompts, below, and create a disc directory of the appropriate size required for the number of files specified. Sectors 1 through 31 of the disc are loaded from a file "CMSBOOTxxxxx" which may be located on any on-line disc. The xxxxx characters are ignored by the utility; only the leading seven characters are compared when the on-line discs are searched (that is, the file specification, searched for is equivalent to CMSBOOT=). The discs are searched in descending channel order, cartridge discs first, then a further scan of Burroughs Super Mini Disks.

PROMPTS AND RESPONSES

The function will request the necessary input by means of prompt messages as detailed below. Invalid input will cause the prompt to be repeated.

<u>PROMPT</u>	<u>INPUT FORMAT</u>	<u>REMARKS</u>
DRIVE	3 characters	The drive unit containing the disc to be initialised DFA, DKA, etc.
<p>Note: At this point the function will perform the disc surface test. The sector address of each track is displayed in binary upon the keyboard indicators (PK24 = 1, PK23 = 2, PK22 = 4 etc.). The disc surface test will take several minutes, the exact time depending upon the type of disc being initialised.</p>		
DATE	MM/DD/YY	Eight characters, for example 01/01/76
FILES	Up to 3 numerals less than 255	The maximum number of files which the disc is to contain. This number determines the number of sectors to be allocated as the disc directory.
SERIAL	6 numerals	The user's identifying serial number.
PACK	Up to 7 characters	The name by which the disc will be known to the system, that is, the < disc-id >.
OWNER	Up to 14 characters	Any identifying character string.
RESTRICTED	Y or N	This option is included for future CMS implementations. The normal response is N.

OUTPUT MESSAGES

The following messages may be output by the utility during the execution of the function.

Normal Execution

The message

NO. OF BAD SECTORS: < integer >
is displayed when a disc has been successfully initialised.

Normal Termination

The message

END IN

is displayed, and the Function Select state is entered, whenever the function is discontinued. This will be described as "the function ENDS".

Errors During Execution

The following messages are displayed in the event of the corresponding error.

1) TRACK 0 BAD.

This message is displayed if any sector in Track 0 cannot be used. A CMS disc contains reserved information in Track 0 (for example, the warmstart bootstrap). The function will END.

2) Dxy DEVICE ERROR.

This message is displayed if a hardware error is encountered during IN.

x identifies the type of disc: F—Burroughs Super Mini-disc; K—Cartridge Disc.

y identifies the unit: A, B, C, etc.

The message will be followed by one of the following displays:

DRIVE INOPERABLE
WRITE INHIBITED
OFF CYLINDER
SEEK TIMEOUT
CONTROLLER PROBLEM.

DRIVE INOPERABLE may be caused if the associated disc is made not ready accidentally.

WRITE INHIBITED indicates that the disc in the specified drive has its write lockout indicator set. Ensure that the write lockout hole (Burroughs Super Mini-disc) is covered, or the write lockout plug (Cartridge Disc) is flush with the surface of the disc cartridge.

All other messages indicate a malfunction of the drive unit. The customer confidence routine (see CUSTOMER CONFIDENCE ROUTINE) should be executed, and field engineering should be notified.

3) LOAD CMS BOOTSTRAP FILE AND HIT OCK1 TO CONTINUE.

This message is displayed if no file in the group CMSBOOT= can be found on any disc on the system. A suitable file may be LD'ed to any on-line disc, of a disc, or a disc containing a suitable file may be installed in an available drive.

4) BOOTSTRAP VERSION <bootstrap version string> USED

This message is displayed when the bootstrap file has been successfully written.

LD (Load Disc)

——LD— < disc—id > —FROM— < mfid > ——

This function will load all files contained on the dump tape identified by < mfid > to the disc identified by < disc—id >. A dump tape is one produced by the DUMP or UNLOAD functions of the BIL utility "LD" (which executes under MCP control). Each sector written to disc is verified.

OUTPUT MESSAGES

The following messages will be output by the utility during the execution of the function.

Normal Execution

The message

< file—id > LOADED

is displayed for each file loaded and verified.

If the dump tape is a reel from a multi-reel dump, then the message

LOAD REEL < integer >

will be displayed when the next reel is required. This message will be repeated if the reel number specified by < integer > is incorrect.

Normal Termination

The message

END LD

is displayed when the function terminates. The utility ends the Function Select state. This will be described as "the function ENDS".

Abnormal Termination

The message

FUNCTION ABORTED

is displayed if a hardware error prevents proper execution of the function. The utility enters the Function Select state. This will be described as "the function ABORTS".

Error During Execution

The following messages will be output by the utility in the event of the corresponding error.

1) NOT DUMP TAPE.

This message is displayed, and the function ENDS, if the tape identified by < mfid > is not a correctly formatted dump tape.

2) < mfid > NOT PRESENT.

This message is displayed, and the function ENDS, if the tape identified by < mfid > is not installed and ready.

3) UNRECOVERABLE CASSETTE ERROR.

This message is displayed if an error is encountered whilst attempting to read the tape. The function will END. This error may be caused by accidentally opening the cassette drive unit.

4) PACK < disc—id > NOT FOUND

This message will be displayed if the specified output disc is not installed and ready. The function will END.

5) O/P DISC NOT WRITE PERMIT.

This message is displayed if the specified output disc is write protected. Ensure that the write lockout hole (Burroughs Super Mini-disc) is covered, or that the write lockout plug (cartridge disc) is flush with the surface of the cartridge. The function will END.

6) DUPLICATE FILE NAME— < file—id >.

This message is displayed if a file of the specified name already exists in the directory of the destination disc. The function will continue with the next file on the tape.

7) CANNOT ALLOCATE AREAS FOR— < file—id >.

This message is displayed if there is no appropriately sized available area on the destination disc for the specified file. The function will continue with the next file on the tape.

8) O/P ERROR— < file—id >.

This message will be displayed if a disc write error is encountered whilst loading the file identified by < file—id >. The function will continue with the next file on the tape.

9) AREA SIZES TOO SMALL FOR— < file—id >.

This message will be displayed if a multi-area file cannot be allocated areas of a suitable size, the function will continue with the next file on the tape.

10) NAME LIST FULL.

This message is displayed if no space is left in the directory of the destination disc to take the name of the specified file. The function will END.

11) Dxy DEVICE ERROR.

This message is displayed if a hardware error is encountered on the destination disc.

x identifies the type of disc: F—Burroughs Super Mini-disc; K—Cartridge Disc.

y identifies the unit: A, B, C, etc.

This message will be followed by one of the following displays:

DEVICE INOPERABLE
OFF CYLINDER
SEEK TIMEOUT
CONTROLLER PROBLEM.

These messages indicate a malfunction of the specified drive unit. The customers confidence routine (see CUSTOMER CONFIDENCE ROUTINE) should be executed, and field engineering should be notified. Note that DRIVE INOPERABLE will be displayed if the specified drive is accidentally made Not Ready.

LS (List Size)

———LS— < disc—id > ——

This function will list the file-names and corresponding file-sizes (in sectors) of the files on the disc identified by < disc—id >.

OUTPUT MESSAGES

The following messages may be output by the utility during the execution of the LS function.

Normal Execution

The message

< file—id > SIZE < integer >

is displayed for each file found in the directory of the specified disc. If no files are found, the utility will produce no output. The < file—id > and < integer > entries identify the file-name and number of sectors respectively.

Normal Termination

The message

END LS

will be displayed when the function terminates. The utility will enter the Function Select state. This will be described as “the function ENDS”.

Abnormal Termination

The message

FUNCTION ABORTED

will be displayed if a hardware error prevents proper execution of the function. The utility will enter the Function Select state. This will be referred to as “the function ABORTS”.

Errors During Execution

The following messages will be output by the utility in the event of the corresponding errors.

- 1) PACK < disc—id > NOT ON LINE.
This message is displayed if the specified disc is not installed or ready. The function will END.
- 2) Dxy DIRECTORY I/O ERROR.
This message is displayed if a read or write error is encountered whilst attempting to access the directory of the specified disc. The directory structure of the disc may be corrupted.

x identifies the type of disc: F—Burroughs Super Mini-disc; K—Cartridge Disc.
y identifies the unit: A, B, C, etc.
- 3) Dxy DEVICE ERROR.
This message is displayed if a hardware error is encountered during the execution of the function. Dxy is as specified in 2 above. The message will be followed by one of the following messages:

DRIVE INOPERABLE
WRITE INHIBITED
OFF CYLINDER
SEEK TIMEOUT
CONTROLLER PROBLEM.

DEVICE INOPERABLE may be caused if the specified disc is made not ready accidentally.
WRITE INHIBITED indicates that the disc in the specified drive has its write lockout indicator set. Ensure that the Write Lockout Hole (Burroughs Super Mini-disc) is covered, or that the Write Lockout Plug (Cartridge Disc) is flush with the surface of the cartridge.

All other messages indicate a malfunction of the drive unit. The customer confidence routine (see CUSTOMER CONFIDENCE ROUTINE) should be executed, and field engineering should be notified.

OL

OL (Print Status of ^{All} ~~Cassette~~ Drives)

— OL —

This function will display a one line message for each (potential) cassette drive on the system, giving the status of the respective drive.

OUTPUT MESSAGES

For each cassette drive, one of the following messages is printed to indicate its current state:

< file-name > ON CTx
NO CASSETTE ON CTx
STAND ALONE CASSETTE ON CTx
UNLABELLED CASSETTE ON CTx

The function terminates with message:

END OL

——RF—— < disc—id > ——

This function provides all the facilities of IN, except the disc recording surface test. A CMS label, and a CMS directory structure are written to the disc. Any information previously contained on the disc will be lost. The disc label will contain parameters supplied by the responses to the appropriate prompts, below, and the directory will be of the minimum size required for the number of files specified. The original parameters are displayed to record the change and to assist re-input of the same data when required. Sectors 1 through 31 are not altered.

PROMPTS AND RESPONSES

The function will request the necessary input by means of prompt messages as detailed below. Invalid input will cause the prompt to be repeated.

<i>PROMPT</i>	<i>INPUT FORMAT</i>	<i>REMARKS</i>
DATE	MM/DD/YY	Eight characters, for example 01/23/77.
FILES	Up to 3 numerals Less than 255	The maximum number of files which the disc is to contain. This number determines the number of sectors to be allocated as the disc directory.
SERIAL	6 numerals	The user's identifying serial number.
PACK	Up to 7 characters	The name by which the disc will be known to the system, that is, the < disc—id >.
OWNER	Up to 14 characters	Any identifying character string.
RESTRICTED	Y or N	This option is included for future CMS implementations. The normal response is N.

OUTPUT MESSAGES

The following messages may be output by the utility during the execution of the function.

Normal Execution

The function will display the current values of the fields which may be changed prior to allowing keyboard input to change the fields. The display is in the following format.

```
DATE      < mm/dd/yy >
FILES     < integer >
SERIAL    < integer >
PACK      < disc—id >
OWNER     < character-string >
RESTRICTED < Y > or < N >.
```

Normal Termination

The message

END RF

is displayed, and the Function Select state is entered, whenever the function is discontinued. This will be described as "the function ENDS".

Abnormal Termination

The message

FUNCTION ABORTED

is displayed, and the Function Select state is entered, if a hardware error prevents proper execution of the function. This will be described as "the function ABORTS".

Errors During Execution

The following messages will be displayed in the event of the corresponding error.

1) PACK < disc—id > NOT ON LINE.

This message is displayed if the specified disc cannot be found. The function ENDS.

2) Dxy DIRECTORY I/O ERROR.

This message is displayed if a read or write error is encountered whilst attempting to access the directory of the specified disc. The directory structure of the disc may be corrupted.

x identifies the type of disc: F—Burroughs Super Mini-disc; K—Cartridge Disc.

y identifies the unit: A, B, C, etc.

The function ABORTS.

3) Dxy DEVICE ERROR.

This message is displayed if a hardware error is encountered during RF. Dxy is as specified in 2 above. The message will be followed by one of the following displays:

DRIVE INOPERABLE
WRITE INHIBITED
OFF CYLINDER
SEEK TIMEOUT
CONTROLLER PROBLEM.

DRIVE INOPERABLE may be caused if the specified drive is accidentally made not ready.

WRITE INHIBITED indicates that the disc in the specified drive has its write lockout indicator set. Ensure that the write lockout hole (Burroughs Super Mini-disc) is covered, or the write lockout plug (cartridge disc) is flush with the surface of the cartridge.

All other messages indicate a malfunction of the drive unit. The customer confidence routine (see CUSTOMER CONFIDENCE ROUTINE) should be executed, and field engineering should be notified.

—————RL— < disc—id > —————

This function allows the identity of any CMS disc to be changed. The function will prompt the operator to supply the new identity.

PROMPT AND RESPONSE

The function will display the message
PACK

and expect the new disc identity to be input. Up to seven characters may be entered as the new disc identity.

OUTPUT MESSAGES

The following messages may be output by the utility during the execution of the function.

Normal Termination

The message

END RL

is displayed, and the Function Select state is entered, whenever the function is discontinued. This will be described as “the function ENDS”.

Abnormal Termination

The message

FUNCTION ABORTED

is displayed, and the Function Select state is entered, if a hardware error prevents proper execution of the function. This will be described as “the function ABORTS”.

Errors During Execution

The following messages will be displayed in the event of the corresponding error.

1) PACK < disc—id > NOT ON LINE.

This message is displayed if the specified disc cannot be found. The function ENDS.

2) Dxy DIRECTORY I/O ERROR.

This message is displayed if a read or write error is encountered whilst attempting to access the directory of the specified disc. The directory structure of the disc may be corrupted.

x identifies the type of disc: F—Burroughs Super Mini-disc; K—Cartridge Disc.

y identifies the unit: A, B, C, etc.

The function ABORTS.

3) Dxy DEVICE ERROR.

This message is displayed if a hardware error is encountered during RL. Dxy is as specified in 2 above. The message will be followed by one of the following displays:

DRIVE INOPERABLE
WRITE INHIBITED
OFF CYLINDER
SEEK TIMEOUT
CONTROLLER PROBLEM.

DRIVE INOPERABLE may be caused if the specified drive is accidentally made not ready.

WRITE INHIBITED indicates that the disc in the specified drive has its write lockout indicator set. Ensure that the write lockout hole (Burroughs Super Mini-disc) is covered, or the write lockout plug (cartridge disc) is flush with the surface of the cartridge.

All other messages indicate a malfunction of the drive unit. The customer confidence routine (see CUSTOMER CONFIDENCE ROUTINE) should be executed, and field engineering should be notified.

```

RM < disc-id > / < file-id >
   |
   | < disc-id > / < group-id >
   |

```

This function will remove a file, or group of files from disc. The disc from which files are to be removed is identified by < disc-id >. A single file identified by < file-id >, or a group of files identified by < group-id > may be removed; that is, the space occupied by the specified file(s) is returned to the available table, and the name(s) are deleted from the directory. Note that to remove all files from a disc, it may be faster to "RF" the disc rather than to use the format "RM < disc-id > / =", if the disc contains a large number of files.

OUTPUT MESSAGES

The following messages may be output by the utility during the execution of the function.

Normal Execution

The following message is displayed for each file successfully removed:

< file-id > REMOVED

Normal Termination

The message

END RM

is displayed, and the utility enters the Function Select state, when execution of the function is discontinued. This is described as "the function ENDS".

Abnormal Termination

The message

FUNCTION ABORTED

is displayed and the Function Select state is entered, if a hardware error prevents proper execution of the utility. This is described as "the function ABORTS".

Errors During Execution

The following messages will be displayed in the event of the corresponding error.

1) PACK < disc-id > NOT ON LINE.

This message is displayed, and the function ENDS, if the specified disc is not located.

2) < file-id > NOT FOUND.

This message is displayed if the specified file cannot be found. If a group remove is in progress, the function will proceed with the next file, otherwise the function ENDS.

3) Dxy DEVICE ERROR.

This message is displayed if a hardware error is encountered on the specified drive.

x identifies the disc type: F—Burroughs Super Mini-disc; K—Cartridge Disc.

y identifies the unit: A, B, C, etc.

The message will be followed by one of the following displays:

DRIVE INOPERABLE
 WRITE INHIBITED
 OFF CYLINDER
 SEEK TIMEOUT
 CONTROLLER PROBLEM.

DRIVE INOPERABLE may be caused if the specified disc is accidentally made not ready.

WRITE INHIBITED indicates that the disc in the specified drive has its write lockout indicator set. Ensure that the write lockout hole (Burroughs Super Mini-disc) is covered, or that the write lockout plug (cassette disc) is flush with the surface of the disc cartridge.

All other messages indicate a malfunction of the drive unit. The customer confidence routine (see CUSTOMER CONFIDENCE ROUTINE) should be executed, and field engineering should be notified.

—————WS—————

This function causes execution control to be passed to the disk load permanent ROM routine. Refer to WARM START PROCEDURE later in this section for full details of warm start.

WARM START PROCEDURE

The B80 CMS Warm Start Procedure is the means by which a B80 CMS Master Control Program (MCP) on disc may be made operational. Warm Start may be initiated as described below, or via the WS function of the Stand-Alone Utility, see STAND ALONE UTILITY.

INITIATING WARM START

The following procedure must be followed to place the system under MCP control.

- 1) Perform the bootstrap load procedure using a B80 CMS disc containing the Master Control Program, file identity "MCP" (and the file "SYSCONFIG", if extended memory is in use). The bootstrap load is fully described in BOOTSTRAP LOAD, but the salient features are:
 - Apply power to the system (and disc drives)
 - Install the disc and ready the unit
 - Press PK2

The system should halt with the keyboard indicators associated with PK's 3 through 6 illuminated.

- 2) Press PK3. PK3 is the leftmost blue key below the *illuminated* keyboard indicators, that is, the third key from the left in the row of blue keys. The Warmstart bootstrap will search the disc directory of the disc from which the bootstrap was loaded for a system file called "MCP". If no such file is found, the ROM search for a disc containing bootstrap is resumed, starting with the next lower address disc (that is, the top drive of the same channel, or the bottom disc of the next lower channel, as applicable). When an MCP file is found, the relevant portions of MCP code are loaded to main memory (RAM) and execution control is passed to MCP.
- 3) Enter the current date when requested. The MCP will display a message of the form:
B-80 MCP VERSION < version—number > < Julian—date >, followed by the current status of all disc units on the system. The system disc will be presented last.

The message

ENTER DATE AS MM/DD/YY

is displayed.

Press the Ready Request key (the rectangular blue button above the PK lights) and enter the current date in the format specified. Leading zeros are optional. Terminate with any OCK. The MCP will display the date in conventional and Julian forms, and identify the day of the week. The MCP idle loop will be entered, and the system will be ready for use.

Possible Errors

The following errors may be encountered whilst loading the Stand-Alone Utility.

- 1) The sequential extinguishing of keyboard indicators does not occur.
 - The MTR switch (behind the processor door, above and to the right of the power on switch) is in the MTR position. Transfer the MTR switch to NORMAL and press Load Enable.
- 2) PK2 is ignored.
 - The keyboard is locked in "shift" mode. Press the shift key below the shift-lock key, and press PK2 again.
- 3) The numeric light illuminates instead of PK3 through PK6.
 - PK1 was pressed. Press Load Enable then PK2.

- 4) The keyboard error light illuminates, and a keyboard light pattern is displayed when PK2 is pressed.
 - If the lights D1 through D8 are illuminated, no bootstrap was found. Check that a correct disc is installed and ready, press Load Enable, and select PK2 again.
 - If one of the lights D1 through D8 is extinguished, a disc error was encountered whilst attempting to locate a bootstrap. The *extinguished* D light indicates the channel number of the disc in error. D1 corresponds to channel 0, D8 corresponds to channel 7. Other keyboard lights indicate relevant information as shown in the table below. Take a note of the light pattern for field engineering use. Power off the faulty disc and replace with a backup copy. Press Load Enable, then PK2.
- 5) The keyboard error light illuminates, and a keyboard light pattern is displayed when PK3 is pressed.
 - Bootstrap has been loaded from a disc, but MCP could not be loaded from that disc. This may be caused if a disc error exists, or if no disc on the system contains an MCP file. The D light which is *illuminated* indicates the channel number of the disc in error. D1 corresponds to channel 0, D8 corresponds to channel 7. Other keyboard lights indicate relevant information as shown in the table below. Note that both a bootstrap and an MCP must be present on the same disc—this is a safeguard to prevent the MCP of another CMS implementation being accepted by Warm Start. If a suitable disc is present on the system, take a note of the light pattern for field engineering use. Power off the disc and replace with a backup copy. Press Load Enable, PK2, and PK3.

Note: Errors 4 and 5 above may be caused by a disc drive fault. This may be checked by attempting to warm start the disc in another drive. If the disc successfully warm starts, run the Customer Confidence Routine (see CUSTOMER CONFIDENCE ROUTINE) and notify field engineering.

- 6) All keyboard indicators alternatively illuminate and extinguish.
 - PK4 or PK5 was pressed during warmstart, and the system has just been powered up. Random Access Memory contains parity errors which have not been deleted by execution of MCP. Press Load Enable, PK2, and PK3.
- 7) The Warm start stops with D2 illuminated. The error light is not illuminated.
 - The serial printer is not ready. Check that the forms transport is properly closed, and the printer cover is properly latched. The variable forms spacing lever (60 cps printer) may be unlatched. The Warm Start will continue when the fault has been corrected. It may be necessary to press Ready followed by OCK1 with 180 cps printers.
- 8) The system may print one of the following messages when displaying the status of Burroughs Super Mini-disc units:

DKX < disc—id > SHOULD BE RE-INITIALISED SOON
DKX < disc—id > BAD DISK—CANNOT BE LOADED.

These messages indicate respectively that 16 or 32 bad sectors were identified on the specified disc. All required information should be copied from the disc to a new, initialised disc. The second message should not be encountered in normal use if notice is taken of the first message. Receipt of the second message may be caused by mistreatment of discs, for example, handling the exposed recording surface.

Table of Keyboard Indicators

Channel Indicators

D1—Channel 0	D5—Channel 4
D2—Channel 1	D6—Channel 5
D3—Channel 2	D7—Channel 6
D4—Channel 3	D8—Channel 7

If an error is encountered whilst the ROM Disc Load Routine is attempting to locate bootstrap, the D light corresponding to the channel in error will be extinguished. All other D lights will be illuminated. The PK lights have the significance shown below. If all D lights are illuminated, no bootstrap was found, and the PK lights have no significance.

If an error is encountered whilst the bootstrap is attempting to locate MCP, the D light corresponding to the channel in error will be illuminated. All other D lights will be extinguished. The PK lights have the significance shown below.

Primary Status Indicators

- PK1—Off indicates top drive; On indicates bottom drive
- PK2—Off indicates SEEK COMPLETE
- PK3—Off indicates END OF CYLINDER
- PK4—Off indicates SEARCH COMPLETE
- PK5—Off indicates SECONDARY STATUS Condition (below)
- PK6—On indicates OPERATIONAL
- PK7—Off indicates SEEK INCOMPLETE
- PK8—On indicates GOOD STATUS (will not be seen).

Secondary Status Indicators

- PK9 —On indicates EQUAL
- PK10—On indicates ON CYLINDER
- PK11—Off indicates ILLEGAL SEEK
- PK12—On indicates WRITE INHIBIT
- PK13—Off indicates SECTOR NOT FOUND
- PK14—Off indicates LRC ERROR (Parity)
- PK15—Off indicates ILLEGAL COMMAND SEQUENCE
- PK16—Off indicates DEVICE ERROR.

These indicators are only significant if SECONDARY STATUS (PK5 above) is OFF.

Retry Count Indicators

The indicators associated with PK17 through PK24 indicate the number of retries attempted before an error is declared.

MEMORY DUMP

This facility is provided in bootstrap to allow the contents of Random Access Memory to be written to cassette or to a file on disc. A memory dump should be taken immediately if the system enters the Bootstrap Load State (PK1 and PK2 lit) spontaneously (that is, not in response to a PO of the system disc), or if Load Enable has to be pressed to "unlock" the system. Note that the contents of memory will be destroyed if PK1 is pressed in the Bootstrap Load State, or if the machine power is removed. The utility PMB80 (refer to the CMS Master Control Program (MCP) Reference Manual, form number 2007555) may be used to produce an analysed listing of the contents of the cassette or disc file.

The Memory Dump will destroy the contents of the following locations:

- 4096 (@ 1000 @)—4110 (@ 100E @)
- 4118 (@ 1016 @)—4127 (@ 101F @)
- 4144 (@ 1030 @)—4159 (@ 103F @)
- 4208 (@ 1070 @)—4223 (@ 107F @)
- 4225 (@ 1081 @)
- 8192 (@ 2000 @)—9517 (@ 252D @).

Memory Dump to Cassette

The contents of Random Access Memory may be dumped to cassette as detailed below. The cassette will be given the identity "MEMDUMP/MEMORY".

- 1) Perform the bootstrap load procedure using any B80 CMS disc. The bootstrap load is fully described in **BOOTSTRAP LOAD**, but the salient features are;
 - Press Load Enable (Note—power off will corrupt memory)
 - Install a suitable disk and ready the unit
 - Press PK2

The system should halt with the keyboard indicators associated with PK's 3 through 6 illuminated.

- 2) Press PK4.
The numeric light will be illuminated.
- 3) Insert a Write Enabled cassette (that is, the red tabs cover the write enable holes) in a cassette drive unit. Ensure that the supply reel is to the left. If the tape is not fully rewound, wait until the tape has rewound, and is halted at clear leader. Press the numeric or alphanumeric numeral key which corresponds to the drive unit containing the tape. Cassette unit 1 is the rightmost unit. Any key except the numerals 1 through 4 will be ignored. The selected cassette will be labelled **MEMDUMP/MEMORY**, the contents of Random Access Memory will be written to the tape in 256 byte blocks, and an ending label will be written. During the dump, the memory address being dumped is displayed on the PK9 through PK16 keyboard indicators in increments of 256 bytes. At the end of the dump, PK17 through PK24 indicators are illuminated. Press Load Enable and Warm Start the system to continue operation (see **INITIATING WARM START**).
- 4) PK's 3 through 6 will be illuminated for further selection (for example—warm start MCP).

Possible Errors

The following errors may be encountered during a Memory Dump.

- 1) The sequential extinguishing of keyboard indicators does not occur.
 - The MTR switch (behind the processor door, above and to the right of the power on switch) is in the MTR position. Transfer the MTR switch to **NORMAL** and press Load Enable.
- 2) PK2 is ignored.
 - The keyboard is locked in "shift" mode. Press the shift key below the shift-lock key, and press PK2 again.
- 3) The numeric light illuminates instead of PK3 through PK6.
 - PK1 was pressed. A memory dump must not be taken if PK1 is accidentally pressed. Press Load Enable and warmstart the system (see **INITIATING WARM START**).
- 4) The keyboard error light illuminates, and a keyboard light pattern is displayed when PK2 is pressed.
 - If the lights D1 through D8 are illuminated, no bootstrap was found. Check that a correct disc is installed and ready, press Load Enable, and select PK2 again.
 - If one of the lights D1 through D8 is extinguished, a disc error was encountered whilst attempting to locate a bootstrap. The extinguished D light indicates the channel number of the disc in error. D1 corresponds to channel 0, D8 corresponds to channel 7. Other keyboard lights indicate relevant information as shown in the table below. Take a note of the light pattern for field engineering use. Power off the faulty disc and replace with a backup copy. Press Load Enable, then PK2.

The system warmstarts instead of illuminating the numeric light.

—PK3 was pressed. A memory dump must not be taken if PK3 is accidentally pressed, as portions of memory will be overwritten by MCP. Allow the warmstart to complete.

All keyboard indicators alternatively illuminate and extinguish when PK4 is pressed.

—The system has just been powered up, and Random Access Memory contains parity errors which have not been deleted by execution of MCP. A memory dump is therefore pointless at this time. Press Load Enable to exit from the condition.

- 7) The keyboard error light illuminates when a numeral is pressed.
 - The wrong cassette drive has been selected.

- The cassette is Write Inhibited.
 - The tape is rewinding.
 - Correct the fault, press Reset, then the correct numeral.
- 8) The keyboard error light illuminates after approximately ten seconds.
- The cassette is installed backwards (that is, the supply reel is on the right). Install the tape correctly, press Reset, then the correct numeral.
- 9) The keyboard error light illuminates during the dump.
- An unrecoverable tape error has been encountered. This may be caused by accidental opening of the cassette drive unit. Install a new tape (or allow the tape to rewind if the drive was accidentally opened), press Reset, then the correct numeral.

Table of Keyboard Indicators

Channel Indicators

D1—Channel 0	D5—Channel 4
D2—Channel 1	D6—Channel 5
D3—Channel 2	D7—Channel 6
D4—Channel 3	D8—Channel 7

If an error is encountered whilst the ROM Disc Load Routine is attempting to locate bootstrap, the D light corresponding to the channel in error will be extinguished. All other D lights will be illuminated. The PK lights have the significance shown below. If all D lights are illuminated, no bootstrap was found, and the PK lights have no significance.

If an error is encountered whilst the bootstrap is attempting to locate MCP, the D light corresponding to the channel in error will be illuminated. All other D lights will be extinguished. The PK lights have the significance shown below. Refer to Error 5 above.

Primary Status Indicators

- PK1—Off indicates top drive; On indicates bottom drive
- PK2—Off indicates SEEK COMPLETE
- PK3—Off indicates END OF CYLINDER
- PK4—Off indicates SEARCH COMPLETE
- PK5—Off indicates SECONDARY STATUS Condition (below)
- PK6—On indicates OPERATIONAL
- PK7—Off indicates SEEK INCOMPLETE
- PK8—On indicates GOOD STATUS (will not be seen).

Secondary Status Indicators

- PK9 —On indicates EQUAL
- PK10—On indicates ON CYLINDER
- PK11—Off indicates ILLEGAL SEEK
- PK12—On indicates WRITE INHIBIT
- PK13—Off indicates SECTOR NOT FOUND
- PK14—Off indicates LRC ERROR (Parity)
- PK15—Off indicates ILLEGAL COMMAND SEQUENCE
- PK16—Off indicates DEVICE ERROR.

These indicators are only significant if SECONDARY STATUS (PK5 above) is OFF.

Retry Count Indicators

The indicators associated with PK17 through PK24 indicate the number of retries attempted before an error is declared.

Memory Dump to Disc

The contents of Random Access Memory may be dumped to disc as detailed below. The contents of Random Access Memory will be written into an existing file of identity MEMDUMP which will be searched for on the disc from which bootstrap was loaded.

- 1) Perform the bootstrap load procedure using a B80 CMS disc containing an empty file of identity MEMDUMP. (The utility GEN.DUMPFL may be used to create a suitable empty file.) The bootstrap load is fully described in BOOTSTRAP LOAD, but the salient features are:
 - Press Load Enable (Note—power off will corrupt memory)
 - Install the disk and ready the unit
 - Press PK2

The system will halt with the keyboard indicators associated with PK's 3 through 6 illuminated.

- 2) Press PK5.
The disc from which bootstrap was loaded will be searched for a file of identity MEMDUMP. The contents of Random Access Memory will be copied into this file.
- 3) PK's 3 through 6 will be illuminated for further selection (for example, warm start MCP).

Possible Errors

The following errors may be encountered whilst taking a memory dump.

- 1) The sequential extinguishing of keyboard indicators does not occur.
 - The MTR switch (behind the processor door, above and to the right of the power on switch) is in the MTR position. Transfer the MTR switch to NORMAL and press Load Enable.
- 2) PK2 is ignored.
 - The keyboard is locked in "shift" mode. Press the shift key below the shift-lock key, and press PK2 again.
- 3) The numeric light illuminates instead of PK's 3 through 6.
 - PK1 was pressed. A memory dump must not be taken if PK1 is accidentally pressed. Press Load Enable and warm start the system (see INITIATING WARM START).
- 4) The keyboard error light illuminates, and a keyboard light pattern is displayed when PK2 is pressed.
 - If the lights D1 through D8 are illuminated, no bootstrap was found. Check that a correct disc is installed and ready, press Load Enable, and select PK2 again.
 - If one of the lights D1 through D8 is extinguished, a disc error was encountered whilst attempting to locate a bootstrap. The *extinguished* D light indicates the channel number of the disc in error. D1 corresponds to channel 0, D8 corresponds to channel 7. Other keyboard lights indicate relevant information as shown in the table below. Take a note of the light pattern for field engineering use. Power off the faulty disc and replace with a backup copy. Press Load Enable, then PK2.
- 5) The keyboard error light illuminates, and a keyboard light pattern is displayed when PK5 is pressed.
 - Bootstrap has been loaded from a disc, but no file MEMDUMP was found. This may be caused if a disc error exists, or if no disc on the system contains a MEMDUMP file. The D light which is *illuminated* indicates the channel number of the disc in error. D1 corresponds to channel 0, D8 corresponds to channel 7. Other keyboard lights indicate relevant information as shown in the table below. Note that both a bootstrap and a MEMDUMP file must be present on the same disc. If a suitable disc is present on the system, take a note of the light pattern for field engineering use. Power off the disc and replace with a backup copy. Press Load Enable, PK2, and PK5.

Note: Errors 4 and 5 above may be caused by a disc drive fault. This may be checked by using the disc in another drive. If the disc successfully dumps, run the Customer Confidence Routine (see CUSTOMER CONFIDENCE ROUTINE) and notify field engineering.

- 6) The system warmstarts instead of dumping memory.
 - PK3 was pressed. A memory dump must not be taken if PK3 is accidentally pressed, as portions of memory will be over written by MCP. Allow the warmstart to complete.
- 7) All keyboard indicators alternatively illuminate and extinguish when PK5 is pressed.
 - The system has just been powered up, and Random Access Memory contains parity errors which have not been deleted by execution of MCP. A memory dump is therefore pointless at this time. Press Load Enable to exit.

Table of Keyboard Indicators

Channel Indicators

D1—Channel 0	D5—Channel 4
D2—Channel 1	D6—Channel 5
D3—Channel 2	D7—Channel 6
D4—Channel 3	D8—Channel 7

If an error is encountered whilst the ROM Disc Load Routine is attempting to locate bootstrap, the D light corresponding to the channel in error will be extinguished. All other D lights will be illuminated. The PK lights have the significance shown below. If all D lights are illuminated, no bootstrap was found, and the PK lights have no significance.

If an error is encountered whilst the bootstrap is attempting to locate the MEMDUMP file, the D light corresponding to the channel in error will be illuminated. All other D lights will be extinguished. The PK lights have the significance shown below.

Primary Status Indicators

- PK1—Off indicates top drive; On indicates bottom drive
- PK2—Off indicates SEEK COMPLETE
- PK3—Off indicates END OF CYLINDER
- PK4—Off indicates SEARCH COMPLETE
- PK5—Off indicates SECONDARY STATUS Condition (below)
- PK6—On indicates OPERATIONAL
- PK7—Off indicates SEEK INCOMPLETE
- PK8—On indicates GOOD STATUS (will not be seen).

Secondary Status Indicators

- PK9 —On indicates EQUAL
- PK10—On indicates ON CYLINDER
- PK11—Off indicates ILLEGAL SEEK
- PK12—On indicates WRITE INHIBIT
- PK13—Off indicates SECTOR NOT FOUND
- PK14—Off indicates LRC ERROR (Parity)
- PK15—Off indicates ILLEGAL COMMAND SEQUENCE
- PK16—Off indicates DEVICE ERROR.

These indicators are only significant if SECONDARY STATUS (PK5 above) is OFF.

Retry Count Indicators

The indicators associated with PK17 through PK24 indicate the number of retries attempted before an error is declared.

CUSTOMER CONFIDENCE ROUTINE

The Customer Confidence Routine (CCR) is a stand-alone, micro-programmed utility which allows all hardware functions of the B80 system to be exercised. The processor, memory, and all peripheral devices (and associated controllers) are dynamically tested, and a listing is produced identifying areas of potential failure. It is recommended that the routine is executed periodically to identify such areas before actual failure. Note that the routine tests "worst case" conditions, therefore errors which are detected by the routine may be causing no errors in normal operation.

FUNCTIONAL SUMMARY

The initial section of the CCR is stored in an alternate area of Read only Memory (ROM). This area is inaccessible until the System Test Switch labelled MTR, behind the processor door, is transferred. The normal (execution) state of this switch is down; the switch must be moved up to permit execution of the Customer Confidence Routine.

The remaining sections of the routine are stored on cassette or disk (for cassetteless systems). The system will contain a cassette or a disk version of the initial routines in ROM as appropriate.

The ROM portion of the tests determine whether sufficient system resources are available to allow loading from the cassette (or disc). The processor, a portion of Random Access Memory (RAM), and the appropriate device controller are tested. Failures in these areas will be indicated upon the keyboard lights, and will require field engineering analysis. If no faults are found, the cassette (or disc) is read, and control is passed into RAM. The individual cassette or disc programs are automatically read into memory as required, and will be referred to as "overlayable routines".

The overlayable routines provide three levels of testing which are to be used as follows.

- 1) Automatic—Quick Test
Basic confidence test to be executed weekly in order to identify potential trouble areas.
- 2) Automatic—Full Test
A more comprehensive test of the system to be executed when a fault is suspected.
- 3) Manual—Field Engineering Test
Diagnostic tool for use by trained Field Engineers.
The level 3 routines above are not within the scope of this manual.

The overlayable routines communicate with the operator using English language statements, and all input communication is through the keyboard. The keys PK1 through PK8 are used to provide almost all the required input, therefore these keys are tested before any further tests are attempted. The operator will be requested to assist the routine when some action is required which cannot be performed programmatically, for example, installing a disc.

The overlayable routines will perform full tests of the processor and memory, and will test each peripheral controller present in the system. In the event that more than one device of the same type is present, the operator should identify the correct device by the channel number which is printed at the start of each test. A list of all channels and their corresponding peripheral devices is produced before any tests are commenced.

The output listing produced by each test is in the form of three columns. The leftmost column is reserved for narrative displays, and requests for assistance. The central column displays test results and responses. The rightmost column is reserved for error reporting in the form of "FIELD CODES". The presence of a FIELD CODE on the output listing should be reported to the Burroughs Field Engineer. All listings should be kept and presented to the

Burroughs Field Engineer on his next visit, even when no FIELD CODE is present. The listings contain information which will assist the Field Engineer to identify degraded performance before components fail.

REQUIREMENTS

If the system to be tested is a cassetteless system, at least one Write Enabled disc (initialised using the FE Stand-Alone Function—see STAND-ALONE UTILITY) containing the CCR programs should be provided. The CCR programs have identities BDSMTR01xx through BDSMTR24xx where xx is a version number and is not taken as part of the file—id for location purposes (that is, the file searched for is equivalent to—for example “BDSMTR01=”). The disc must be installed in a drive on physical channel 2.

If the system to be tested contains both cassettes and discs *and* contains MTR ROM which expects cassette then a Write Enabled cassette tape version of CCR must be provided. This tape must be installed in a drive on physical channel 2.

For each disc drive which is on the system, the CCR program will expect a Write Enabled FE initialised disc to be installed when the drive is tested. When using the Quick Test facility, the user should, when directed by program printout, install separate discs in each drive in order to minimise operator intervention (if a drive does not contain a disc when the drive is scheduled for testing, the operator will be requested to install a suitable disc). If the Quick Test facility is not being used, a suitable disc should be installed in each drive as directed by the program printout. These extra discs need not contain the BDSMTRxxxx programs. It is possible to transfer the disc on channel 2 as each drive is tested, but the disc should be restored to the original drive when disc testing is finished.

A Write Enabled cassette tape version of CCR should be installed in each cassette drive when requested by the program printout. In the Quick Test mode of operation, all cassette drives to be tested should have suitable cassette tapes installed when requested. Any drive not containing a cassette tape will be skipped in the Quick Test. Drives containing an incorrect (not CCR) tape will generate a Field Code when tested.

Each printer (including the console printer) which is available on the system must be fitted with listing paper of at least 13½" (343mm) printing area (14½" including pin feed perforations). Each Wide Line Printer should be fitted with a format tape loop containing a Channel 1 hole.

If the system is installed in a data communication network, then the operator disconnect data communications plugs should be removed.

CONFIDENCE ROUTINE OPERATION

The confidence routine is mainly automatic in operation, but in addition to loading and initiating the test program, the operator may be required to monitor the system indicator lights and operate the system controls as and when directed by the printout. To execute the Customer Confidence Routine, perform the following procedure.

- 1) Remove all normal system media (discs, cassettes, listing paper) from the system, and install the listing paper, format tapes, and ONE CCR disc OR cassette (as appropriate) in a drive unit on channel 2. Ensure that the console printer left pin feed tractor is set to position zero.
- 2) Insert the PK identification strip associated with the Customer Confidence Routine into the plastic cover which is located over the PK lights. If no such strip is available, refer to the table of PK assignments later in this section.
- 3) Set the System Test Switch to the Test (up) position.
- 4) Press the Load Enable Button. This initiates the ROM portion of the test program. If no faults occur, the second section of the test program will be initiated automatically.

- 5) Ensure that the second section of the test program is initiated within 60 seconds—at the beginning of the second section, the serial printer is initialised (the carrier moves to the right bumper and then to the zero print position), and the program title is printed.
- 6) Perform the first block of instructions which appear on the printout.
- 7) Select "Automatic Test".
- 8) If the basic confidence routine is to be executed, select Quick Test, and install the media detailed in REQUIREMENTS above. The routines will then execute automatically. If insufficient discs are available, then a disc may be moved from drive to drive as requested. Cassette drive and printers which have no media installed will not be tested. If the Full Test is to be executed then the media should be mounted and dismounted as directed by program printout.
- 9) Remove and retain the program printout and report any fault conditions, that is, FIELD CODES, which have been printed on the right hand side of the form. Enter the date and time on the printout as indicated.
- 10) Restore the System Test Switch to the normal (down) position.
- 11) Remove the Customer Confidence Routine media and restore all normal media.

TABLE OF PK ASSIGNMENTS

PK1	— YES
PK2	— CONTINUE
PK3	— NO
PK4	— RESTART (this test)
PK5	— field engineering use
PK6	— CANCEL (this test)
PK7-24	— field engineering use

POSSIBLE ERRORS

The following conditions may prevent proper execution of the Confidence Routine.

- 1) The second section of the test program is not initiated at Step 7 of CONFIDENCE ROUTINE OPERATION.
 - a) the System Test Switch is not in the TEST (up) position
—correct the condition and press Load Enable.
 - b) more than one cassette drive unit contains a cassette (cassette version CCR)
—remove all cassettes except the Confidence Routine cassette, and press Load Enable.
 - c) the cassette is installed backwards (cassette version CCR)
—ensure that the cassette is installed with the supply reel to the left.
 - d) the cassette or disc contains an irrecoverable read error
—install another (backup) copy, and press Load Enable.
 - e) the cassette or disc drive unit has a fault
—if more than one unit is available on channel 2, install the media in an alternative unit, and press Load Enable.
 - f) The serial printer is in an error condition
—ensure that the forms transport is properly closed and that the printer cover is properly latched. The variable forms spacing lever (60 cps printer) may be unlatched. Press Load Enable.

If none of the above conditions apply, it must be assumed that an error has been discovered by the ROM routine. Notify the local Burroughs Field Engineering Representative.

- 2) If, at any time during the execution of the test program, an incorrect response is made (for example, PK1 = YES is pressed accidentally instead of PK3 = NO) then the test may be restarted by pressing PK4. An incorrect response of this type will usually cause a FIELD CODE to be issued if the test is allowed to continue. The FIELD CODE will be incorrect and should be disregarded.
- 3) On systems containing more than one peripheral device of the same type, it is possible for the operator to perform some requested action upon the wrong device. Press PK4 to restart the test.

KEYTOP LEGENDS

The following key is identified by the corresponding description in the Customer Confidence Routine printout.

“ON/OFF KEY” describes the Ready Request key, that is, the rectangular blue key above the PK lights.

SECTION 6

B 800 DEPENDENT ROUTINES

GENERAL

This section describes the routines that are executed on the B 776/B 800 hardware only. These routines will cold start a CMS system disk for B 776/B 800 systems, will warm start an existing CMS system disk, and are dump analyzers for the dump files on disk created by the operating system on B 776/B 800 systems.

CREATE

The CREATE program will load the release tape to create a TOTALDISK. Since the TOTALDISK is also released with the release tape, this routine is only necessary if the release disk (TOTALDISK) can not be used locally due to a read error.

SET-UP PROCEDURE

Before loading the bootstrap loader, perform the following steps to enable loading CREATE:

1. Depress tape unit POWER switch to power up unit.
2. Mount release magnetic tape on a tape drive. Bring tape to BOT marker by depressing the LOAD switch on the tape unit twice in succession.

OPERATING INSTRUCTIONS

To load the bootstrap loader:

1. Depress the LOAD/NORMAL pushbutton on the CPU. It should be illuminated red for load.
2. Depress the system CLEAR pushbutton.
3. Start papertape reading by depressing upper rocker of memory loader switch and inserting paper. The program on papertape is loaded in memory starting at location 0000.
4. After papertape loading is complete, depress the LOAD/NORMAL pushbutton on the CPU to return to normal mode (it should be illuminated white).
5. Depress the system CLEAR pushbutton.
6. Depress the READY pushbutton or the INPUT REQUEST on the operator console.
7. Depress the numeric key whose number corresponds to the tape port, to load CREATE; or the disk port to load DSKGEN. If the port number is greater than 9, depress J, K, or L for port numbers 10, 11 and 12 respectively.

BOOTSTRAP ERROR HANDLING

- a. If a disk error occurs during execution of the bootstrap, the program waits with the disk status in MIR.
- b. If a tape error occurs during execution of the bootstrap, the program waits with the tape status in MIR.
- c. If a fatal hardware error occurs during execution of the bootstrap, the program halts with the error number in MIR. Successive force stepping of the program will place the contents of MPCR and BMAR in MIR.
- d. If a disk error occurs while DSKGEN is loading (system loader) the program waits with the disk status in MIR.
- e. If a fatal error occurs while DSKGEN is loading (system loader) the program waits with the error code in MIR.

Operating in CREATE mode, the disk cartridge on unit 1 of drive 1 will first be initialized. If no bad sectors are found, the DSKGEN segments will be read from tape and written to disk, followed by the RELEAS, FIGTRE, NAMESS, and LETTER files. The actual start and stop disk address of each file will be printed on the operator's interface device when the entire segment or file has been successfully written to disk (see figure 6-1).

```

6
BOJ CREATE

DATE (MMDDYY): 042977
DISK PORT # IS 4
INITIALIZING DISK
INITIALIZE COMPLETE
SEG 63 0012 0016 LOADED
SEG 00 0017 0038 LOADED
SEG 01 0039 003B LOADED
SEG 02 003C 0044 LOADED
SEG 03 0045 004A LOADED
SEG 04 004B 004F LOADED
SEG 05 0050 0057 LOADED
SEG 06 0058 0060 LOADED
SEG 07 0061 0071 LOADED
SEG 08 0072 0074 LOADED
SEG 09 0075 008E LOADED
SEG 0A 008F 0094 LOADED
SEG 0B 0095 0099 LOADED
SEG 0C 009A 00A7 LOADED
SEG 0D 00AB 00AC LOADED
SEG 0E 00AD 00B1 LOADED
SEG 0F 00B2 00B3 LOADED
SEG 10 00B4 00B8 LOADED
SEG 11 00B9 00C5 LOADED
RELEAS 00C6 1FB0 LOADED
FIGTRE 1FB8 1F9D LOADED
NAMESS 1F9E 1FB5 LOADED
LETTER 1FB6 2275 LOADED

EOJ CREATE

```

Figure 6-1. Sample CREATE Printout

ERROR MESSAGES AND RECOVERY

The following is a summary of tape and disk error messages of conditions which can occur while operating CREATE.

TAPE ERROR MESSAGES		
ERROR PRINTOUT	PROBABLE CAUSE	OPERATOR ACTION
TAPE LABEL ERROR C OR P NOT SPECIFIED ERROR DSKGEN FILE ERROR * * DSKGEN INDEX ERROR * * TAPE READ ERROR * * TAPE NOT READY ERROR	File-ID is not "GENSYS" User's portion of label does not contain a C or P in position 81. DSKGEN loader (Seg 63) was not the first DSKGEN segment read from tape. DSKGEN segment · was not between 0 and 22. A Service Too Late, Parity Error, or Tape Error has been detected. Tape unit is not ready to accept a command from the processor.	These errors indicate a tape format error in the release tape Reload program from bootstrap and allow one retry. If unsuccessful, a new tape will have to be used. Reload program from bootstrap

DISK ERROR MESSAGES		
ERROR PRINTOUT	PROBABLE CAUSE	OPERATOR ACTION
BAD SEG - CANNOT BE TOTALDISK	A bad sector has been found during initialization. NOTE: The above error condition may indicate a serious misalignment of the cartridge. Do not attempt more than one retry with the same cartridge, otherwise the read/write heads may be damaged.	1. Re-try CREATE. 2. Try another cartridge. 3. Check for proper functioning of the disk unit.
NO AVAILABLE DISK ERROR × × DISK NOT READY	No more space on disk for files. Disk unit unable to accept commands from processor.	Technical assistance required. Ready disk unit.
ADDR DISK ERROR × ×	1. Bad sector on disk cartridge; Operate-Compare successful but returned address incorrect. 2. Illegal track address.	1. Initialize cartridge. 2. This error usually indicates a firmware or hardware malfunction.
DVCE DISK ERROR × ×	Status error in disk unit which firmware does not recognize. Disk unit, processor or firmware error.	1. Ready unit and retry.
FILE DISK ERROR × ×	1. Disk unit not operational. 2. Write lockout on disk drive. 3. Cable connector to disk unit not properly seated.	1. Ready unit. 2. Enable write on cartridge. 3. Ensure proper connection to disk unit.
PRTY DISK ERROR	During any phase of initialization indicates an LRC read error on the cartridge.	A re-try of the operation may be successful, however, a true LRC error means that the cartridge is no longer usable. First determine that the disk unit is operating properly before attempting retry.
SEEK DISK ERROR × ×	Seek error on disk unit.	Technical assistance required.
SERV DISK ERROR × ×	Service late on disk read or write. Disk unit, processor, or firmware error.	CREATE source may be faulty try again.
TIME DISK ERROR × ×	Instruction timeout on disk unit. Can be caused by hardware malfunction or bad cartridge sector address.	1. Disk cartridge may require initialization. 2. System clear button should be pressed.

DISK GENERATOR

GENERAL DESCRIPTION

- The Disk Generator program (DSKGEN) can perform the following functions:
- Initiate unit 1, drive 2 and build the initial CMS directory.
 - Execute a system generation in either prompting or no-prompting mode.
 - Copy from any drive to any other drive.
 - Warmstart the user system cartridge.
 - Print the release letter on the operator console, or line printer.
 - Replace firmware segments and programs on the user system cartridge.

The individual functions are described in detail in the following sections.

The disk generator program consists of 18 code segments. DSKGEN is microprogrammed for speed and access to all areas of disk. The System Loader resides at track 0, face 1 on TOTALDISK and is read into memory using the bootstrap papertape. After load time, always resident in core are: 256 registers; ACOMMON (which consists of a manager routine, input/output control, read/write routines, accept/display routines, etc.); a disk controller and a SPO or console controller. The overlay region is used for various functions by DSKGEN, including:

- a. Initial setup procedures.
- b. Disk initialization and build CMS directory code.
- c. Disk generation in prompting or no-prompting code.
- d. Copy disk to disk.
- e. Bootstrap to warmstart.
- f. Fatal error routines.
- g. Printing the LETTER file.
- h. Patching the user's system cartridge.

It is important to realize that disk accesses are necessary to perform the functions of DSKGEN. Only remove TOTALDISK at the appropriate times as specified in the operating instructions.

OPERATING INSTRUCTIONS

1. Execute the DSKGEN bootstrap loader.
2. Depress the READY button or the INPUT REQUEST key on the operator console to define to the program which device is on the particular system.
3. Index the numeric port number for the disk unit. The bootstrap will now wait if the disk unit is not ready, then proceed to the DSKGEN code segments.

NOTE

If any error occurs in the above sequence, return to Step 1.

4. DSKGEN will print:
B 700/B 800 SYSTEM DISK GENERATION
ENTER DATE (MMDDYY)
Enter the six-digit date.
DSKGEN will print:
ENTER TIME (HHMM)
Enter the time.

NOTE

Date will be syntaxed for validity and the message repeated if invalid. Time will be syntaxed for valid numerics.

5. DSKGEN will print:
1 INIT
2 DSKGEN
3 COPY
4 BOOT
5 LETTER
6 PATCH
SELECT

Select the option by entering a number 1-6. The message will be repeated for an invalid selection. This is referred to as the SELECT routine throughout the rest of the instructions.

The selected routine Initialize, System Generation, Copy, Bootstrap, Letter, or Patch will now be loaded from disk and begin execution. The routines are described below.

INITIALIZATION ROUTINE (INIT)

Before selecting Option 1, the user must first install a scratch disk cartridge in unit 1, drive 2. The option will then cause the program to print "INITIALIZING DISK" and begin initializing the cartridge. If a disk cartridge is not present, or is not write enabled, this will result in a disk error message and return to the SELECT routine. After the disk is initialized, CMS directory will be built and then return to the SELECT routine after printing any bad segments found (up to 50 bad segments are allowed for a system disk). Other messages should be handled as previously described under disk error messages.

After an CMS disk has been initialized, there will follow a series of displays to the operator for which responses are required to specify disk directory size and cartridge label fields.

Operating procedures for the Question-Answer routine are as follows (responses must be directed to the alphanumeric keyboard if entered on an electronic console):

1. "ENTER NO OF TRACKS FOR DIR (1-9):" will print. Only a numeric response from 1-9 will be allowed, with the resulting directory format given in table 6-1, Directory Format Table.
2. "ENTER SERIAL NUM77 will print. One to six numeric keys (0-9) on the alpha keyboard must be entered.
3. "ENTER PACK ID" will print. One to seven valid alphanumeric characters must be entered.
4. "ENTER OWNERS ID" will print. One to fourteen valid alphanumeric characters must be entered.

Table 6-1. Directory Format Table

Requested Tracks (Sectors)	Available Entries	Directory (File) Entries
1 (32)	116	25
2 (64)	116	55
3 (96)	116	84
4 (128)	116	113
5 (160)	116	143
6 (192)	116	172
7 (224)	116	201
8 (256)	116	231
9 (288)	116	256

SYSTEM DISK GENERATION ROUTINE (DSKGEN)

If Option 2 is selected, the system disk generation procedure will be entered, starting with a Question-Answer routine in which the user can specify which operating mode prompting or no-prompting will be used for the disk generation. In both methods, selections of the subsystems to be loaded to the user's system disk are made by entering the subsystem numbers via the keyboard.

As part of the release procedure, the subsystems are assigned into three groups: choices, exclusive options, and options. The number of allowable selections from any one group of subsystems depends on the type of the subsystem group. When only one subsystem must be selected from the subsystem group, this is a choice. When only one subsystem may be selected from the subsystem group, this is an exclusive option. When any number of subsystems may be selected from the subsystem group, this is an option. The choices, options, and exclusive options are predetermined. In prompting mode, subsystem groupings are printed for the user and each selection is syntaxed as it is made. In no-prompting mode, selections are syntaxed after all entries are complete. The first selection of both prompting and no-prompting is always the selection of the system to be generated.

QUESTION-ANSWER ROUTINE

Operating procedures for the Question-Answer routine are as follows:

1. "RESPOND TO QUESTIONS WITH 1 FOR YES AND 2 FOR NO" will print.
2. "FE MODE" will print. If a user requires a listing of the segment type, segment name, and segment load address, the response should be a 1, otherwise a 2.
3. "DO YOU REQUIRE PROMPTING?" will print. The user should respond 1 or 2.
4. If the user responds YES to prompting, the following will print:

```
PLEASE CHECK DRIVE 2 FOR
SCRATCH DISK PRESENT
DISK DRIVE ON
DISK WRITE ENABLED
ARE DISKS READY?
```

5. When all conditions are true, the user should respond 1. If the user responds NO to the Step 3 prompt, then the message "ARE DISKS READY?" will print. If the scratch disk is present, on, and write enabled, the user should enter a 1.
6. "INITIALIZING DISK" will print. At this time the scratch disk will be initialized. If any errors occur during initialization the appropriate error message will be displayed. The same Question-Answer routine as described for INIT will be entered, requiring the same operator responses as specified therein.

PROMPTING MODE OPERATING INSTRUCTIONS

1. "SELECTIONS ARE MADE BY ENTERING THE 4-DIGIT SUBSYSTEM NUMBERS" will print.
2. System Selection:
 - a. "SELECT SYSTEM:" will print followed by the valid system numbers and system names to be selected.
 - b. "SELECTION 001" will print. The user should respond by entering one of the system numbers. If an invalid selection is made, an appropriate error message will appear. "SELECTION 001" will print. The user should re-enter a valid system number; the previous selection is ignored.
3. Subsystem selection:
 - a. If selection is a choice, "YOU MUST SELECT (ONLY) ONE OF:" will print followed by the group of subsystem numbers and subsystem names to be selected.
 - b. If selection is an exclusive option, "YOU MAY SELECT (ONLY) ONE OF:" will print followed by the group of subsystem numbers and subsystem names to be selected. The last selection to print will be "9999 NO SELECTION".
 - c. If selection is an option, "YOU MAY SELECT ANY NUMBER OF:" will print followed by the group of subsystem numbers and subsystem names to be selected. The last selections to print will be "9998 ALL SELECTIONS" and "9999 END SELECTION".
 - d. "SELECTION XXX" (where XXX = the number) will print. XXX is incremented by one before display.
 - e. If selection is a choice, the user should enter the subsystem number from the group of choices displayed. If an invalid selection is made, the appropriate error message will appear and "SELECTION XXX" (XXX will not change) will print again. The user should re-enter a valid selection; the previous selection is ignored. When a valid selection is entered, the object code for that subsystem is then loaded to disk.
 - f. If selection is an exclusive option, the user should enter a subsystem number from the group of exclusive options displayed or a "9999" if no selection is to be made. If an invalid selection is made the appropriate error message will appear and "SELECTION XXX" (XXX will not increment) will print again. The user should re-enter a valid selection; the previous selection is ignored. When a valid selection is entered, the object code for that subsystem is then loaded to disk.
 - g. If selection is an option, the user should enter the subsystem number(s) from the group of options displayed, or a "9999" if no selection is to be made, or a "9998" if all selections are to be made. If an invalid selection is made, the appropriate error message will appear and "SELECTION XXX" (XXX will not increment) will print again. The user should re-enter a valid selection; the previous selection is ignored. When a valid selection is entered, the object code for that subsystem is then loaded to disk. "SELECTION XXX" (XXX has been incremented by 1) will print. If another selection is to be made, enter the next subsystem number. Duplicate selections will be ignored. When all subsystems have been selected from the group, enter a "9999" to end selections for that group.
 - h. Procedures for subsystem selection will be repeated until all subsystems have been selected. The message "SYSTEM DISK COMPLETE XX:XX" will print, where XX:XX represents the time. "SELECT" will then print, meaning that the user has been returned to the Select routine.

PROMPTING ERROR MESSAGES

"ERROR - NOT A SYSTEM"

The first selection is not a system.

"ILLEGAL SELECTION"

The selection made is an invalid selection for that group of systems. The user should make another selection.

B700 SYSTEM DISK GENERATION
RELEASE 1276

ENTER DATE (MMDDYY) 122976

HAPPY HOLIDAYS

ENTER TIME (HHMM) 1346

- 1 INIT
- 2 DSKGEN
- 3 COPY
- 4 BOOT
- 5 LETTER
- 6 PATCH

SELECT 2

SYSTEM DISK GENERATION 13:46

RESPOND TO QUESTIONS WITH 1 FOR YES 2 FOR NO
FE MODE? 2

DO YOU REQUIRE PROMPTING? 1

SYSTEM WILL BE GENERATED ON DRIVE B

PLEASE CHECK DRIVE B FOR

-SCRATCH DISK PRESENT

-DISK DRIVE ON

-DISK WRITE ENABLED

ARE DISKS READY? 1

INITIALIZING DISK

ENTER NO OF TRACKS FOR DIR (1-9) : 5

ENTER SERIAL NUM 135790

ENTER PACK ID RELB776

ENTER OWNERS ID TIODTN

SELECTIONS ARE MADE BY ENTERING A 4-DIGIT SYSTEM NUMBERS

SELECT SYSTEM:

0057 CMS--SYSTEM

SELECTION 001 0057 CMS--SYSTEM

YOU MAY SELECT ANY NUMBER OF:

0001 MPL2--COMPILER

0013 COBOL--COMPILER

0025 RPG--COMPILER

0036 NDL--COMPILER

0040 PSL--COMPILER

0044 SQMCS--PROGRAM

0046 CMCS--PROGRAM

0048 NDL--SAMPLE

9998 ALL SELECTIONS

9999 END SELECTION

SELECTION 002 0001 MPL2--COMPILER

SELECTION 003 0013 COBOL--COMPILER

SELECTION 004 0025 RPG--COMPILER

SELECTION 005 9999

SYSTEM DISK COMPLETE 13:53

Figure 6-1. Sample DSKGEN, Prompting Mode

NO-PROMPTING MODE OPERATING INSTRUCTIONS

1. Upon entering no-prompting mode, the following instructions will be displayed:
9999 - ENDS SELECTION
9998 - INCLUDES ALL OPTIONS
1ST SELECTION MUST BE A SYSTEM
2. The above information is followed by a display of the heading format to be used in the selection process:
SELECTION · SS-NO DESCRIPTION YES-NO?
3. The system selection process will then commence:
 - a. "001" will print under SELECTION.
 - b. The user should enter the system number of the system to be coldstarted.
 - c. The system name will print under DESCRIPTION and the printing element will proceed to the YES-NO column.
 - d. If the selection is correct as displayed, enter a "1" under YES-NO. If the selection must be changed or corrected, enter a "2" under YES-NO; the printing element will proceed to the next line, print 001 (SELECTION · remains the same), and a new selection can be made (previous selection is ignored).
 - e. Following the entry of a "1" (YES response) the selection will be checked for validity. If an invalid system selection was made, an appropriate error message will be printed and the procedure will return to the "001" selection state. This process will continue until a valid selection is made or the procedure is terminated.
4. Following successful system selection, the procedure then proceeds to subsystem selection:
 - a. The print element returns to the SELECTION column, and the selection number is printed incremented by one.
 - b. Select appropriate subsystem(s):
 - 1) Enter "9998" if all options are to be included; or
 - 2) Enter subsystem selections individually.In either case, the printing element will proceed to the YES-NO column after the selection is made. Enter a "1" if the entry is correct, or a "2" if the subsystem number requires correction. If a "1", the SELECTION will be incremented allowing additional subsystem selections to be made; if a "2", the same selection number will be repeated.
 - c. Terminate the selection routine by entering "9999".
5. The object code for all selected subsystems is loaded to disk, providing that the selection is valid.
6. If an invalid subsystem number is detected, the subsystem number and appropriate error message will print.
7. The message "SYSTEM DISK COMPLETE XX:XX" will print, where XX:XX represents the time in hours and minutes of the day.
8. The message "SELECT" will print, notifying the user that control has been returned to the Select routine.

NO-PROMPTING ERROR MESSAGES

"ERROR - NOT A SYSTEM"

The first selection is not a system.

"ILLEGAL SELECTION"

The selection made is an invalid selection for the system being generated.

"AMBIGUOUS SELECTION - IGNORED"

More than one choice or option of the same group was entered. The second one was ignored.

"XXXX DUP SELECTION - IGNORED"

Duplicate selection, where XXXX is the subsystem number. Identical selections were made. The second one was ignored.

"RQD CHOICE XXX NOT MADE"

Required choice not made, where XXX is the choice number. A required subsystem was not entered. System disk will not be generated.

"SEL TABLE FULL"

Selection table full (100 selections allowed per system generation).

COPY

If Option 3 is selected, the program will print messages prompting the user for the source and destination drive numbers. To return to the SELECT routine enter zero, (0 from the alpha keyboard) for the source drive.

B700 SYSTEM DISK GENERATION
RELEASE 1276

ENTER DATE (MMDDYY) 122976

HAPPY HOLIDAYS

ENTER TIME (HHMM) 1357

- 1 INIT
- 2 DSKGEN
- 3 COPY
- 4 BOOT
- 5 LETTER
- 6 PATCH

SELECT 2

SYSTEM DISK GENERATION 13:57

RESPOND TO QUESTIONS WITH 1 FOR YES 2 FOR NO
FE MODE? 2

DO YOU REQUIRE PROMPTING? 2
ARE DISKS READY? 1

INITIALIZING DISK

ENTER NO OF TRACKS FOR DIR (1-9): 9

ENTER SERIAL NUM 135790

ENTER PACK ID RELB776

ENTER OWNERS ID TIODTN

9999 ENDS SELECTIONS

9998 INCLUDES ALL OPTIONS

1ST SELECTION MUST BE A SYSTEM

SELECTION #	SS-NO	DESCRIPTION	YES-NO?
-------------	-------	-------------	---------

001	0057	CMS--SYSTEM	1
002	0001	MPL2--COMPILER	1
003	0013	COBOL--COMPILER	1
004	0025	RPG--COMPILER	1
005	0036	NDL--COMPILER	2
005	9999		

SYSTEM DISK COMPLETE 14:04

Figure 6-3. Sample DSKGEN, Prompting Mode

In order to prime the CMS destination cartridge label, copy will print the following:

- a. For Copying TOTALDISK:
SOURCE DRIVE A-H (0 TO TERMINATE): A
DESTINATION DRIVE A-H: B
ARE YOU COPYING TOTALDISK? 1
COPYING
SOURCE DRIVE A-H (0 TO TERMINATE): O
- b. For Copying a CMS System Disk:
SOURCE DRIVE A-H (0 TO TERMINATE): A
DESTINATION DRIVE A-H: B
ARE YOU COPYING TOTALDISK? 2
ENTER PACK ID ABCDEFG
ENTER OWNERS ID ABCDEFG12345
COPYING
SOURCE DRIVE A-H (0 TO TERMINATE): O

The expected responses to the "ENTER PACK ID" and "ENTER OWNERS ID" displays are as described for those fields in INIT Question-Answer Routine.

TOTALDISK may be removed during this process but must be replaced before entering "0". A disk error message will result if a disk cartridge is not present or if a disk error occurs. A new cartridge must be initialized before being copied to. A cartridge with any bad sectors cannot be used in COPY.

COPY performs a disk-to-disk copy of the first 203 cylinder of the source disk. A read-after-write check is done for each track. If the validity check fails, a message "COPY ERROR XXXX" will be printed (where XXXX = disk address). Since DSKGEN is limited by the virtual machine, the copy is a full copy on a single density drive and a half copy on a double density drive. This is sufficient to copy TOTALDISK and any new users system disk but should not be used to copy double density data platters.

NOTE

DSKGEN can be used to create copies between single and double density drives of TOTALDISK and the users system cartridge, but it should be realized it is the directory information of the original disk that will govern the limits of the second disk.

BOOTSTRAP WARMSTART

In CMS the system disk may be placed on any unit and drive. When the system disk is ready, the operator will enter the desired drive. The warmstart routine will be read from the specified disk and begin execution. The typical printout is as follows:

```
BOOTSTRAP WARMSTART  
REMOVE TOTALDISK  
PLACE SYSTEM DISK ON DESIRED DRIVE
```

ENTER DRIVE NUMBER (A-H) WHEN READY

NOTE

Unless TOTALDISK is removed before a CMS Warmstart, it will be destroyed.

RELEASE LETTER

GENERAL DESCRIPTION

The release letter which always accompanies a release may be printed for the user's benefit. If the user does not specify a dump to the line printer, the letter is automatically dumped to the console. If the line printer is selected, and an error occurs, "LINE PRINTER ERROR" will print and the program will return to the SELECT routine. At the end of the dump of the letter file, the program will return to the SELECT routine.

OPERATING INSTRUCTIONS

1. "DO YOU REQUIRE A PRINTER LISTING" will print.
2. User responds "1" (YES) or "2" (NO).
3. If YES, "PRINTER PORT " will print.
4. User should enter printer port number. The line printer should be ready.
5. If NO, the letter will begin printing on the console.

PATCH

GENERAL DESCRIPTION

The function of the patching capability is to replace firmware segments of code files, and programs on a user's system cartridge. The user selects the segments, files, or programs that he wishes to replace. There are two restrictions when patching code segments: (1) The selected segment to be patched must be present on the user's system cartridge and (2) there must be enough disk area to load the patched segment into the same disk area as the selected segment. When patching programs and files, if the selected program or file is present on disk, that program or file will be purged first and then the patched program or file will be loaded. If the selected program or file is not present on disk, the patched program or file will be loaded to disk. The user should be aware that if the disk space cannot be found for the patched program or file, it will not be loaded and the previous program or file has been purged. This is a very important aspect when patching subsystems of programs and files; for example a compiler, because some programs and files of that subsystem possibly will not be loaded. In some cases, the selected program or file cannot be purged, therefore the user's disk is left unchanged and a message is displayed to inform the user.

There are two modes of patching, prompting and no-prompting. In the prompting mode, a list of segments, subsystems, programs, and file names along with their appropriate subsystem number are displayed for the system to be patched. In the no-prompting mode no list is displayed. Selections are made by entering the 4 digit subsystem number. In both modes the first selection must be a system selection.

OPERATING INSTRUCTIONS

1. Upon entering patch the following instructions will be displayed:

RESPOND TO QUESTIONS WITH 1 FOR YES 2 FOR NO
SELECTIONS ARE MADE BY ENTERING 4-DIGIT SYSTEM NUMBERS
DO YOU REQUIRE PROMPTING?

2. Enter a 1 (yes) or 2 (no)
3. If a 1 is entered:
 - a. The group of system names and their system numbers will be displayed.
 - b. "SELECT SYSTEM:" will print followed by the heading format below which is to be used in the selection process.
SELECTION · SS-NO DESCRIPTION YES-NO?
 - c. "001" will print under selection .
 - d. The user should enter the system number of the system to be patched.
 - e. The system name will print under DESCRIPTION and the printing element will proceed to the YES-NO column.
 - f. If the selection is correct as displayed, enter a "1" under YES-NO. If the selection must be changed or corrected, enter a "2" under YES-NO; the printing element will proceed to the next line, print 001 (SELECTION · remains the same), and a new selection can be made (previous selection is ignored).
 - g. Following the entry of a "1" (YES response) the selection will be checked for validity. If an invalid system selection was made, an appropriate error message will be printed and the procedure will return to the "001" selection state. This process will continue until a valid selection is made or the procedure is terminated.
 - h. All segment, subsystem, program, and file names of the selected system along with their appropriate subsystem number will be displayed followed by the message "9999 ends selections".
4. If a 2 is entered:
 - a. The following instructions will be displayed:

1ST SELECTION MUST BE A SYSTEM
9999 ENDS SELECTIONS

- b. The above information is followed by a display of the heading format to be used in the selection process:
SELECTION · SS-NO DESCRIPTION YES-NO?
- c. "001" will print under SELECTION .
- d. The user should enter the system number of the system to be patched.
- e. The system name will print under DESCRIPTION and the printing element will proceed to the YES-NO column.

- f. If the selection is correct as displayed, enter a "1" under YES-NO. If the selection must be changed or corrected, enter a "2" under YES-NO; the printing element will proceed to the next line, print 001 (SELECTION · remains the same), and a new selection can be made (previous selection is ignored).
 - g. Following the entry of a "1" (YES response), the selection will be checked for validity. If an invalid system selection was made, an appropriate error message will be printed and the procedure will return to the "001" selection state. This process will continue until a valid selection is made or the procedure is terminated.
5. Following successful system selection, the procedure then proceeds to the selection routine:
 - a. The print element returns to the SELECTION · column, and the selection number is printed incremented by one.
 - b. The user should enter the subsystem number of the program, file, subsystem, or segment to be patched.
 - c. The printing element will proceed to the YES-NO column after the selection is made. Enter a "1" if the entry is correct, or a "2" if the subsystem number requires correction. If a "1", the SELECTION · will be incremented allowing additional subsystem selections to be made; if a "2", the same selection number will be repeated.
 - d. The selection will then be syntaxed for validity and patchability. If patchable and valid, the patched segment, program or file will replace the present code on disk and a message will be displayed to inform the user. If the selection is not valid or patchable, the appropriate message will be displayed to inform the user.
 - e. Terminate the selection routine by entering "9999". The message "SELECT" will print, notifying the user that control has been returned to the select routine.

ERROR MESSAGES AND RECOVERY

FATAL ERROR MESSAGES

The occurrence of a fatal error condition while operating DSKGEN will cause a branch to the fatal error routine, resulting in an error alarm, an error printout and display, and a program halt.

Recovery may be possible from some fatal error conditions by attempting to re-run the program; however, if recovery fails after repeated re-tries (one re-try for hardware errors), the operator has no recourse other than to obtain technical assistance. When a fatal error condition is detected, the system will respond in the following manner:

1. The FATAL error routine segment will be loaded from TOTALDISK to program memory if not already resident there.
2. The console ERROR indicator will be turned on.
3. The console audible alarm will be sounded.
4. The "D" indicator bank will display the binary value of the error type.
5. The appropriate error message will be printed on the console.

HARDWARE FATAL ERRORS

The console message format for a hardware fatal error condition is:

(REASON FOR ERROR)	A A A A	B B B B	C C C C	D D D D
Error Message	Error	MIR Contents	MPCR Contents	BMAR Contents

A typical fatal error message format is as follows:
 NAND ADDRESS ERR 8005 9001 2020 007B

"0" INDICATOR PATTERN	ERROR MESSAGE	DESCRIPTION
00	MEMORY ERROR	
01	LOADER ADDRESS ERR	Hardware has detected an error in papertape during read-in of data.
02	LOAD ADDRESS ERR	Hardware has detected an addressing error while loading papertape; that is, an attempt was made to load beyond physical confines of memory.
03	LOAD MPM PARITY ERR	Hardware has detected a parity error while loading papertape.
04	NANO PARITY ERROR	Hardware has detected a parity error in an MPM word while attempting to read a nano instruction.
05	NANO ADDRESS ERROR	Hardware has detected an addressing error while attempting to read a nano instruction; that is, an attempt was made to read outside the physical confines of the nano memory.
06	MPM PARITY ERR	Hardware has detected a parity error while attempting to access a micro instruction.
07	MPM ADDRESS ERROR	Hardware has detected an addressing error while attempting to fetch a micro instruction; that is, an attempt was made to execute a micro instruction outside the physical confines of the memory.
09	DPM WRITE ERR	Hardware has detected a memory address in excess of the memory limit register setting while the processor was attempting a write operation.
0A	DPM PARITY ERR	Hardware has detected a parity error while the processor was attempting to read data memory.
0B	DPM READ ERR	Hardware has detected a memory address in excess of the memory limit register setting while the processor was attempting a read operation.
0D	WRITE STEAL ERR	Hardware has detected a memory address in excess of the memory limit register setting while one of the direct memory access channels was attempting to write to memory.
0E	STEAL PARITY ERR	Hardware has detected a parity error while one of the direct memory access channels was attempting to read data memory.
0F	READ STEAL ERR	Hardware has detected a memory address in excess of the memory limit register setting while one of the direct memory access channels was attempting to read from memory.

SOFTWARE FATAL ERRORS

The console error message format for software fatal errors is as follows:

(REASON FOR ERROR)	AAAA	BBBB	CCCC	DDDD
Error Message (See listing next page)	Error #	MIR Contents	MPCR Contents	BMAR Contents

FIB Description

EEEE	FFFF	GGGG	HHHH	IIII	JJJJ	KKKK
Key	Max. Key	Current Page	Records/ Page	Segs/ Page	D-Words/ Record	Start Address
LLLL	MMMM	NNNN	OOOO	OOOO	OOOO	OOOO
Unit	D-Words/ Segment	Total D-Words	----- Data -----			

Example:

```
INVALID KEY      0002 0001 1796 1002
0001 0064 FFFF  005A 0004 0004 0598
0000 005A 0168  0000 0000 0005 9808
```

Software fatal error messages having the above format includes:

ERROR MESSAGE	DESCRIPTION
ILLEGAL - I/O	Interpreter has detected an illegal request to an I/O device.
INVALID KEY	An attempt was made to read an item record with a disk key beyond the file limits.

In addition to containing the above formatted information, the software fatal error messages listed below also include FIGTRE record information following the data field. The format is:

<p style="text-align: center;"> PPPP PPPP PPPP PPPP ----- FIGTRE Record ----- </p>
--

Example:

```
SEG NOR ERROR 0004 2064 1467 175A
0031 FFFF FFFF 0020 0010 0020 0000
0000 005A 0AEO  0000 0000 0000 0008
0000 0000 0000  0000
```

ERROR MESSAGE	DESCRIPTION
INVALID TYPE	FIGTRE file contains a record type.
SEG HDR ERROR	Segment header error. Invalid header type has been detected.
NO PROGRAM SEG	No program segment. An attempt was made to load a program segment not on disk.
POSITIONING	Carrier stall was detected while positioning; carrier overspeed was detected while printing; or the interpreter detected an attempt to position the carrier beyond the limit imposed by the platen size value of the keyboard dataset currently in use by the interpreter.
STACK OVERFLOW	Occurs when choice option stack (max. 20 entries) or substyles stack (max. 4 entries) exceeds limit.

Disk Error Messages

ERROR MESSAGE	DESCRIPTION	OPERATOR ACTION
ADDR DISK ERROR ***	<ol style="list-style-type: none"> 1. Bad sector on disk cartridge; Operate-Compare successful but returned address incorrect. 2. Illegal track address. 	<ol style="list-style-type: none"> 1. Initialize cartridge. 2. This error usually indicates a firmware or hardware malfunction.
DVC DISK ERROR ***	Status error in disk unit which firmware does not recognize. Disk unit, processor, or firmware error.	Ready unit and retry.

ERROR MESSAGE	DESCRIPTION	OPERATOR ACTION
FILE DISK ERROR * * *	<ol style="list-style-type: none"> 1. Disk unit not operational. 2. Write lockout on disk drive. 3. Cable connector to disk unit not properly seated. 	<ol style="list-style-type: none"> 1. Ready unit. 2. Enable write on cartridge. 3. Ensure proper connection to disk unit.
INIT DISK ERROR * * *	More than 50 bad segments or 9 bad areas have been found while initializing disk cartridge. This condition may also be caused by a malfunctioning disk drive.	<ol style="list-style-type: none"> 1. Retry initialization. 2. Try another cartridge. 3. Check for paper functioning of the disk unit.
PRTY DISK ERROR * * *	During any phase of initialization indicates an LRC read error on the cartridge.	A retry of the operation may be successful, however, a true LRC error means that the cartridge is no longer usable. First determine that the disk unit is operating properly before attempting retry.
SEEK DISK ERROR * * *	Seek error on disk unit.	Technical assistance required.
SERV DISK ERROR * * *	Service late on disk read or write. Disk unit, processor, or firmware error.	Create source may be faulty - try again.
TIME DISK ERROR * * *	Instruction timeout on disk unit. Can be caused by hardware malfunction or bad cartridge sector address.	<ol style="list-style-type: none"> 1. Disk cartridge may require initialization. 2. System CLEAR button should be pressed.

Other Error Messages

The following error messages may appear while operating DSKGEN:

ERROR MESSAGE	DESCRIPTION
DIRECTORY MESSAGE * * *	Indicates no available entries in user's protected directory for program date segment. IO-data or utility load.
BAD SEGMENT: XXXX	Indicates that a bad segment was detected where XXXX is the disk address.
NOT FOR INTERPRETER USE	There are bad sectors in the protected area, therefore, this disk cannot be a system platter.
COPY ERROR XXXX	Indicates read-after-write check failure, where XXXX is the disk address.
LINE PRINTER ERROR	Indicates that the line printer is in a Not Ready condition.
TRACK 0 BAD - DISK UNUSABLE	CMS - Bad sectors in necessary disk area.
NO ROOM FOR DIRECTORY - DISK UNUSABLE	CMS - An area on disk cannot be found to contain the disk directory of the size specified during initialize.

CMS System Disk Layout

CMS system disk segments are divided into the following sections:

Disk Segments	Use
0	Label information
1	Security Information
2 thru 9	B 80 Bootstrap Area
0A thru 11	B 1700 Bootstrap Area
12 thru 18	B 700/B 800 Bootstrap Area
19 thru 1F	Reserved for Operating System
20 thru END	Directory Space Disk File Areas

WARM START

The warm start procedure enables an operator to define the system configuration. The system is placed in a normal operating status at the conclusion of a warm start.

The bootstrap loader (COMMON BOOT paper tape) is loaded in the same manner as in CREATE and DSKGEN up to the point of entering a port number. The entry is now the disk drive number A through H which contains the CMS system disk with a B 700MCP file.

WARM-START OPERATING PROCEDURE

NOTE

If an error condition is indicated at any time when performing the following procedure, refer to information under WARM START ERROR HANDLING, table 6-3, for error definitions and operation actions.

All entries are made on the operator console. Proceed as follows:

1. Warm start will print the messages WARMSTART, SYSTEM PACK ID IS pack-id, and HARDWARE CONFIGURATION.
2. Enter the configuration of the system. A valid entry will consist of a two-character mnemonic (table 6-2), a two-character port number, and a one-character device number. The one-character device number has a range of A-H for all devices except DC, for which the range is 0 or 1. If two device numbers are entered, only the first is used except for a multiqueue device, such as disk. Blanks may be entered at any time. Only 40 characters are valid per entry.

NOTE

If an error has occurred during warm start, one of the error messages will be printed on the operator console. The entire configuration of the system must be reentered.

3. Termination of any entry is made by pressing END OF MESSAGE pushbutton or the OCK1 on the numeric keyboard.
4. If an entry is not complete, and an error has been made, pressing the ERROR or RESET button will allow a recreation of that entry.
5. When configuration is complete, END must be entered to allow warm start to continue.
6. To the question DUMP FILE? a YES or NO must be entered. A YES response will cause a file named DMFIL00 to be created on the system disk cartridge. Also, disk space will be allocated to DMFIL00 equal to the size of the main memory of the system. In the unlikely event of a processor or system firmware failure (for example, Memory Address Error) the MCP will store the contents of main memory to DMFIL00. When the system is restored, see clear start, the operator should execute the utility SYSDUMP. SYSDUMP will convert the dump file to a formatted listing which will be of use to support personnel.

A NO response will remove an existing DMFIL00 from the system disk and return the disk space to the available directory.

7. Message ENTER DATE: is printed on the system console. The expected response is mm/dd/yy, where mm = month, dd = day, and yy = year. The entry may be preceded or followed by blanks, but may not contain blanks.
8. If the real time clock (RT) has been configured into the system, the message ENTER TIME: will print on the system console. Warmstart expects a four character response of hhmm (where hh = hour and mm = minute). The entry may be preceded or followed by blanks, but may not contain blanks.

Table 6-2. Valid Device Mnemonics

VALUE	DESCRIPTION
CT	Cassette Tape
DC	Disk Communication ¹
DF	Mini Disk
DK	Disk Cartridge
KB	Console Keyboard
LP	Line Printer
MT	Magnetic Tape
M8	80-Column Card Reader/Printer/Punch
RT	Real-Time Clock and Event Timer
R8	80-Column Card Reader
R9	96-Column Card Reader
SP	Serial Printer

¹The only valid device numbers are 0 or 1 for this mnemonic.

Warm start also contains the following stand-alone functions:

a. IN (INITIALIZE)

DISK CARTRIDGES MAY BE INITIALIZED BY ENTERING "IN" FOR THE HARDWARE CONFIGURATION DURING A WARMSTART. WARMSTART WILL PRINT:

INIT

ENTER DATE (MMDDYY):

ENTER THE 6-DIGIT DATE AND WARMSTART WILL PRINT:

ENTER DRIVE NO (A-H):

ENTER THE DISK DRIVE NUMBER AND INITIALIZING WILL COMMENCE. AFTER THE ENTIRE CARTRIDGE IS INITIALIZED THEN THE FOLLOWING WILL PRINT:

ENTER NO OF TRACKS FOR DIR (1-9):

AFTER THE ENTRY THEN THE FOLLOWING WILL PRINT:

ENTER SERIAL NUM

ENTER THE SERIAL NUMBER AND THE NEXT REQUEST IS:

ENTER PACK ID

UP TO 7 CHARACTERS MAY BE ENTERED, THEN THE NEXT REQUEST IS:

ENTER OWNERS ID

UP TO 14 CHARACTERS MAY BE ENTERED. THE CMS DIRECTORY IS THEN ESTABLISHED ON THE DISK CARTRIDGE. WARMSTART WILL THEN ASK FOR THE HARDWARE CONFIGURATION.

b. CO (COPY)

CMS CARTRIDGES MAY BE COPIED BY ENTERING "CO" FOR THE HARDWARE CONFIGURATION DURING A WARMSTART. TOTALDISK MAY NOT BE COPIED BY THIS COPY, ONLY BY THE COPY FUNCTION IN DSKGEN. WARMSTART WILL PRINT:

COPY: ENTER SOURCE DRIVE (A-H):

AFTER THIS IS ENTERED THEN THE NEXT LINE IS:

ENTER DESTINATION DRIVE (A-H):

WHEN THIS IS ENTERED IT WILL ASK:

ENTER PACK ID

FINALLY IT WILL REQUEST:

ENTER OWNERS ID

WHEN THE COPY IS COMPLETED, WARMSTART WILL ASK FOR THE HARDWARE CONFIGURATION.

c. PA (PATCH)

A FIRMWARE PATCH FUNCTION IS NOW AVAILABLE THROUGH WARMSTART. THIS IS THE ONLY TIME A PATCH MAY BE ENTERED TO THE SOFTWARE. IT CAN BE USED TO UPDATE ANY OPERATING SYSTEM OR INTERPRETER SEGMENT VIA THE OPERATOR CONSOLE.

THE PATCH FUNCTION IS INVOKED BY ENTERING "PA" AS THE FIRST ENTRY OF THE HARDWARE CONFIGURATION. THE KEYBOARD IS ENABLED FOR THE ENTRY OF PATCHES. A PATCH CONSISTS OF ONE OR MORE LINES OR ENTRIES. EACH ENTRY IS TERMINATED BY EOM OR OCK1 (NUMERIC KEYBOARD ONLY). THE PATCH FUNCTION IS TERMINATED BY KEYING IN "END" AT WHICH POINT WARMSTART IS RESTARTED.

Possible error messages and reasons:

1. PO1 PATCH FORMAT ERROR	A. Number of Dwords not in 01-06 range B. Format error detected by checking slash positions C. Correct check digit not present
2. PO2 PATCH SEGMENT ERROR	A. Segment to be patched not present in MCP file B. Patch history segment not present (should not occur)
3. PO3 PATCH CODE ERROR	A. Current code for the patch does not compare with actual code in the segment
4. PO4 PATCH HISTORY NOT RECORDED	A. Over 89 patches have been recorded so there is no more room in the history file. The patch has still been made on disk, but not recorded.

In addition to patching the appropriate segment on disk, a record of all patches is kept in a patch history file.

RESTART PROCEDURE

The restart procedure provides the means to reinstate or restore the system firmware and MCP in memory following any abnormal condition that may have affected the integrity of the system in memory, and also after the system is powered on. Restart reinitiates the system as it had been configured by the most recent warm start.

RESTART OPERATING INSTRUCTIONS

Perform the following procedure to restart the system:

NOTE

Refer to warm start error handling for any errors which may occur.

1. Warm start will print the messages WARMSTART and HARDWARE CONFIGURATION.
2. Enter END on the operator console to request the hardware configuration by the most recent warm start.
3. A successful restart places the system in a system idle state.

NOTE

The dump file, DMFIL00, will remain in the same state as the most recent warm start.

4. ENTER DATE: will be printed on the system console. The expected response is mm/dd/yy where mm = month, dd = day, and yy = year. The entry may be preceded or followed by blanks, but may not contain blanks.
5. If the real-time clock (RT) has been configured into the system, the message ENTER TIME: will print on the system console. start expects a four-character response of hhmm (where hh = hour and mm = minute). The entry may be preceded or followed by, but may not contain blanks.

CLEAR START PROCEDURE

Clear Start will be invoked by the MCP when a condition demands the abrupt termination of all tasks active in the system. The may normally invoke Clear Start by pressing the processor clear button. However, this should be avoided, if at all possible, during a probable disk access. In the process of the Clear Start the MCP will:

1. Create a copy of system memory in the system dump file, DMFIL00, if it exists. (Refer to warmstart procedure). Regardless of the existence of DMFIL00, if a memory parity error is the termination cause, no dump of system memory is performed.

2. Display to the operator console:
 - a. A termination cause.
 - b. A set of MCP registers, current mix No. and current slice.
 - c. The contents of the processor registers MIR, BMAR, and MPCR.
3. Request the operator to input the date and time (if a clock was identified in warm start).
4. Re-establish the system integrity (for example, load system firmware from disk, disk directories are validated . . .)
5. Display the MCP banner and enter system idle.

MCP invoked Clear Starts are of three classes: Central processing Unit failures, virtual memory failures, MCP failures. The CPU failures result in the following termination messages:

<p>MEMORY ERROR</p> <p>MPM PARITY</p> <p>DPM PARITY</p> <p>STEAL PARITY</p> <p>NANO PARITY</p>	<p>There is a borderline error in the memory. The field engineer should be alerted so that he may run the memory test. Being borderline, it may take some time to locate.</p> <p>There is a memory parity error in the location indicated by the contents of MPCR.</p> <p>There is a memory parity error in the location indicated by the contents of BMAR.</p> <p>There is a memory parity error, location unknown.</p> <p>There is nano parity error. The exact name can be determined by:</p> <ol style="list-style-type: none"> 1. Execute the "HEX" option of SYSDUMP. 2. Locate the contents of the word indicated by MPCR. 3. The lower 10 bits indicate address of the nano with bad parity.
--	---

Virtual memory failures may be distinguished by the following messages:

VIRTUAL MEMORY: INSUFFICIENT LOCK SPACE

The memory required for the system firmware and the resident code and data segments of all the active tasks is greater than the physical memory of the system.

VIRTUAL MEMORY: INSUFFICIENT OVERLAY SPACE

Given this set of tasks the unused memory in the system is less than the memory required by a user's overlayable code or data segment.

VIRTUAL MEMORY: PERMANENT ERROR

The virtual memory routines have encountered a disk error in the process of accessing disk for a segment overlay. An additional display is made to aid in the resolution of the error:

VMIO DESCRIPTOR + X0 X1 X2 . . . X23

Xn are hexadecimal digits. The fields within this string which may be of use to the operator are:

X3: = Read, 1 = Write 3 = Write with Read After Write.

4 = Firmware read, 6 = Directory search

X5: Type of error.

0001 = Parity

0010 = Timeout/seek

0011 = Address

X6,X7: Lower eight bits of hardware DDP Status

X12...X15: Disk Address in hexadecimal

X16...X19: Length in 16 bit words

Termination messages which will usually indicate MCP failures are:

<p>**NANO ADDRESS**</p> <p>**MPM ADDRESS ERROR**</p> <p>**DPM ADDRESS READ**</p> <p>**DPM ADDRESS WRITE**</p> <p>**STEAL ADDRESS READ**</p> <p>**OS TASK HAD BAD DATA**</p>	<p>This refers to a nano address error and may occur due to a bad B 700MCP file on the system cartridge, a bad read from disk, or an MCP failure. At the very least system support will require a HEX version of SYSDUMP; the formatted dump would also be helpful.</p> <p>This error is provided for but due to the minimum memory requirements of a B 776 user it will never occur.</p> <p>The contents of BMAR can be used to confirm the correctness of either of these errors. The discussion of reasons and steps to take outlined in NPM ADDRESS ERROR also apply for these two messages.</p> <p>This indicates a direct memory access device, that is, DCP or disk, has accessed a word beyond the memory limits. At the very least, system support will need a formatted version of SYSDUMP for either of these errors.</p> <p>This message will have been preceded by a message with the event number 70, 71, 72, or 73.</p> <p>This normally implies an MCP failure and a formatted dump should be taken.</p>
---	--

Examples of CLEAR/START printouts follow:

a. CLEAR/START

```

**DPM PARITY ERROR**
DMFIL00 NOT CREATED
CURRENT MIX NO. = 0001
CURRENT SLICE = 001B
MIR = 0005
BMAR = 2D67
MPCR = 0130
ENTER DATE: 11/21/75
ENTER TIME: 1045
MCP2. 0 R ##### 75326

```

This indicates a memory parity error at location 2D67 while task 1 was executing. As indicated earlier, memory parity errors do not create a DMFIL00.

b. CLEAR/START

```

DMFIL00 CREATED
CURRENT MIX NO. = 0010
CURRENT SLICE = 0020
MIR = 0000
BMAR = BE2C
MPCR = 0000
ENTER DATE: 11/21/75
ENTER TIME: 1110
MCP 2.0 R ##### 75326

```

Since both MIR and MPCR = 0000 this indicates that the operator probably pressed the clear button on the processor control panel.

NOTE

It is strongly recommended that the clear button not be pressed when the system has active tasks since in process disk write operations may be corrupted.

c. CLEAR/START

```

DMFIL00 CREATED
CURRENT MIX NO. = OS TASK 000E
CURRENT SLICE = 0040
MIR = 4C09
BMAR = BDE4
MPCR = 0000
ENTER DATE: 11/21/75
ENTER TIME: 1200
MCP 2.0 R ##### 75326

```

Since MPCR = 0000, this cannot be a hardware trap to location 0. Since MIR = 0000 this cannot be a result of the clear button. The conclusion is that there has been an MCP failure. The operator should take a HEX version of SYSDUMP.

d. CLEAR/START

```
** DPM ADDRESS READ **
DMFIL00 CREATED
CURRENT MIX NO. = 0001
CURRENT SLICE = 0022
MIR = 2030
BMAR = F3B5
MPCR = 0131
ENTER DATE: 11/21/75
ENTER TIME: 1231
MCP 2.0 R ##### 75326
```

BMAR is certainly out of range, therefore, it is likely to be an MCP failure. Take a HEX dump.

e. CLEAR/START

```
VIRTUAL MEMORY: PERMANENT ERROR
DMFIL00 CREATED
CURRENT MIX NO. = 0001
CURRENT SLICE = 002C
VM10 DESCRIPTOR = 0000014028000272005A2234
ENTER DATE: 11/21/75
ENTER TIME: 1335
MCP 2.0 R ##### 75326
```

This indicates that Virtual Memory has encountered a parity error while reading sector -0272-. X5 indicates a parity error for software.

SYSTEM DUMP ANALYZER (SYSDUMP)

The System Dump Analyzer is a debugging aid for the programmer. In order to operate SYSDUMP the system dump file DMFIL00 must be present on the system disk. Space is reserved for it at warm start time if the operator responds yes ("Y") to DUMP FILE?. Any system dump file which was previously on disk is removed. During restart the system dump file previously on disk is left unharmed. The dump file is created during the system clear start routine which may be initiated by (1) the operating system when it detects a fatal error, or (2) by the operator who may initiate a hard clear by pushing the Clear button on the processor. SYSDUMP examines this file and arranges it into an easily readable format.

While the system is operating, the command to execute the dump is given via the system console. If an unformatted dump is desired, HEX is entered with the execute command.

After a formatted dump, SYSDUMP will go to EOJ. After an unformatted dump, the program will display the message FORMATTED DUMP?. If a formatted dump is desired, the alphabetic character Y is entered via an AX command. SYSDUMP will terminate automatically after the formatted dump. If a formatted dump is not desired, the alphabetic character N is entered via the AX command. The character N, in response to FORMATTED DUMP?, causes SYSDUMP to go to EOJ.

OPERATING INSTRUCTIONS

The System Dump Utility may be executed at any time the system is operating by entering via the operator console:

```
SYSDUMP    HEX
```

HEX is an option that, if entered, will produce an unformatted memory dump. Default is a formatted dump. After an unformatted dump, the program will display:

```
FORMATTED DUMP?
```

A yes (Y) or no (N) answer is expected. After a formatted dump, the program will go to end-of-job (EOJ).

If, during the execution of the formatted dump, the program detects a virtual memory address error (due usually to the timing of the creation of DMFIL00), the formatted dump will be discontinued and replaced by an unformatted dump and the message:

```
PROCEEDING WITH UNFORMATTED DUMP
```

TABLE 6-3. WARM START ERROR MESSAGES

MESSAGE	CAUSE
<p>W00 BAD MCP FILE</p> <p>W01 PORT ALREADY WARMSTARTED</p> <p>W02 INVALID PORT</p> <p>W03 PORT NOT ON 1 TO 12</p> <p>W04 DEVICE ALREADY WARMSTARTED</p> <p>W05 INVALID MNEMONIC</p> <p>W06 INVALID DISK PORT</p> <p>W07 DISK NOT WARMSTARTED</p> <p>W08 TWO DISK ENTRIES</p> <p>W09 DISK DEVICE NOT A TO H</p> <p>W10 DC NOT ON 0 OR 1</p> <p>W11 TOO MANY ENTRIES</p> <p>W12 DISK NEVER WARMSTARTED</p> <p>W13 NO DDP</p> <p>W14 NO DEVICE</p> <p>W15 DUMP FILE SIZE TOO SMALL</p> <p>W16 INVALID</p> <p>TROUBLE X1X2 X3X4X5X6 X7X8X9X10</p>	<p>The contents of the B 700MCP file are corrupt and as such are unusable by warmstart.</p> <p>Two different devices warmstarted to one port.</p> <p>Invalid Port number.</p> <p>Port number greater than 12.</p> <p>Two entries of the same device with the same device number.</p> <p>The mnemonic is not one specified in Table 2-1.</p> <p>The port number for the disk entry is not 4.</p> <p>Disk not entered when warmstarting.</p> <p>The disk has been specified twice during the input of the hardware configuration.</p> <p>Disk Device number not in the range of A to H.</p> <p>Data Communications not an 0 or 1 for device number.</p> <p>More than 40 characters entered.</p> <p>A restart has been attempted before this disk has ever been warmstarted.</p> <p>No DDP number entered.</p> <p>No device number entered.</p> <p>In the process of a restart, the size allocated to the DMFIL00 is less than that required for the size of memory.</p> <p>An invalid date or time entry has been made.</p> <p>Where X1 thru X10 are hexadecimal digits the trouble message will occur when warmstart has detected a hardware fatal error. X3X4X5X6 will be the contents of processor MPCR register at the time the error occurred. X7X8X9X10 will be the address contained in the processor register BMAR. X1X2 is the error code and may be decoded as follows. (Refer to Clear Start procedure for additional details).</p>
<p>80 MEMORY ERROR</p> <p>84 NANO PARITY</p> <p>85 NANO ADDRESS</p> <p>86 MPM PARITY</p> <p>87 MPM ADDRESS</p> <p>89 DPM ADDRESS WRITE</p> <p>8A DPM PARITY</p> <p>8B DPM ADDRESS READ</p> <p>8D STEAL ADDRESS WRITE</p> <p>8E STEAL PARITY</p> <p>8F STEAL ADDRESS READ</p> <p>HELP XX X</p> <p>X is a hexadecimal digit. If XXE00, a failure has occurred when trying to read or write to the disk. The B 700MCP file cannot be found.</p> <p>The eight binary bits in hexadecimal digits xx are numbered as: 7 6 5 4 3 2 1 0</p> <p>Bit Setting:</p> <p>1 = Write Inhibit set.</p> <p>2 = Seek on the drive not working.</p> <p>3 = Illegal address error.</p> <p>4 = File not operational.</p> <p>5 = Seek incomplete.</p> <p>6 = Parity Error.</p> <p>7 = Time Error (Operation Timed Out).</p>	

SYSDUMP ERROR MESSAGES

CANNOT OPEN FILE - SYSDUMP TERMINATING

The operating system detected an error, such as file not present, while trying to open DMFIL00.
DMFIL00 BAD DUMP FILE OR HAS NOT BEEN CREATED

The flag which specifies that DMFIL00 has been created is not set.

CANNOT OPEN FILE B 700MCP - SYSDUMP PROCEEDING WITH UNFORMATTED DUMP

The operating system has detected an error while trying to open the driver file. An unformatted dump will be produced.

ERROR DETECTED - FILE B 700MCP

A disk error has been detected during a read of the file B 700MCP. If an unformatted dump has already been created during this run of the program, SYSDUMP will terminate. Otherwise an unformatted dump will be produced.

ERROR DETECTED - DUMP DRIVER

A disk error has been detected during a read from the driver file. Unless already created, an unformatted dump will be produced.

EOF DETECTED DUMP DRIVER

End of file has been detected during an attempted read of the dump driver. Unless already created, an unformatted dump will be produced.

ERROR DETECTED - DMFIL00

A disk error has been detected during a read from DMFIL00. SYSDUMP will immediately terminate.

EOF DETECTED - DMFIL00

End of file has been detected during an attempted read of DMFIL00. SYSDUMP will immediately terminate.

PROCEEDING WITH UNFORMATTED DUMP

SYSDUMP has detected a bad operator in the driver table and is unable to complete the formatted dump.

EXPLANATION OF FORMATTED DUMPS

This section explains and gives examples of a formatted dump analysis listing in the actual order listed.

The following paragraphs explain the individual sections of the analysis.

HEADING

The format and content of the first page heading section of the printout will be as follows:

B 7 7 6 O P E R A T I N G S Y S T E M D U M P A N A L Y Z E R	
04/29/77	00:00:00
ANALYZER VERSION:	1.0(10/01/76)
OPERATING SYSTEM VERSION:	02/28
DUMP FILE CREATION DATE:	04/29/77

PATCH HISTORY

The program lists the patch number and segment number of each patch made to the MCP or Data Comm Firmware files. This section of the printout will be as follows:

PATCH HISTORY	
PATCH NO	SEG NO
0001	004F
0001	004F
0002	002D
0002	0003
0003	0065
0003	0003
0003	0003
0006	0014

HARDWARE REGISTERS

This section of the printout will be as follows:

PAGE	BMAR	MPCR	ERROR
0000	8049	0000	0000

The hardware registers are identified as follows:

MIR	Memory Information Register.
BMAR	Memory Address Register.
MPCR	Micro Program Count Register.
ERROR	Hardware Error Code from the External Bus.

OPERATING SYSTEM REGISTERS

This section of the printout will be as follows:

OS	FLAGS	MAT	TID	SLICE	MAD	VM	SCL	O/C	SPO	EOJ	TASK/	WSB	ERROR
0080	0000	8049	0000	0040	0140	FFFF	FFFF	FFFF	FFFF	FFFF	8000	0000	0000

BVM	PSEUDO	MICRO	SLICE	DC	TVM	LAST	VM	HOLE	HOLE	SLICE	SDT
1888	1888	102F	1E44	8005	A44D	0000	BE98	0370	102F	0018	805B

LOADER REGISTERS

0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------

ISCT

0000	BFE2	BFCF	BFD9	BFA9	0000	BFC2	BFEB	BF3	0000	0000	0000	0000
------	------	------	------	------	------	------	------	-----	------	------	------	------

The operating system registers are identified as follows:

OS Flags

CURRENT PTRS

HAT
TID
SLICE
MAD

LOCKS

VM
SCL
O/C
SPO
A/D

DMAC-ERR COUNT

BVM

EXTENTS

PSEUDO
MICRO
SLICE
DC

TVM

LAST ALLOC

VM OBJECT

HOLE SIZE

HOLE ADDR

SLICE FLAGS

SDT BASE

LOADER REGISTERS

ISCT

MIX ATTRIBUTES

Flags for use by operating system routines.

Address of the current task's entry in the Mix Attribute table.
Task-ID of currently (at the time of creation of DMFIL00) processing task.
Slice number of currently executing slice.
The last Micro Address Descriptor processed by the operating system.

Flags used by the operating system to protect the non-reentrant code of:

Virtual Memory.
SCL Slice.
Open/Close.
SPO or operator console.
Allocate/Deallocate.

Register used internally to the DMAC-Disk Controller to count retries on a particular Disk I/O.
Address of Beginning of Virtual Memory.

Address of the first D-word after pseudo-nucleus.
Address of the first D-word after micro slice area.
Address of the first D-word after slice area.
Address of the first D-word of data comm area.

Total number of bytes of Virtual Memory.
Address of end of last allocated segment in overlayable area.

Address of object descriptor (slice or segment).
Size of hole required for VM access.
Address of hole required for VM access.
Flags used by the operating system for VM access.

Address of base of SDT.
Workspace reserved exclusively for the loader.
Base address of Interrupt Scan Control Table.

Lists each currently scheduled task by mix number, gives the current state of the task (runnable, short waited, suspended - swappable, suspended - swapped out) and its wait key. If the task is not marked runnable, this column provides the reason for the suspension. This section of the printout will be as follows:

14	SHORT WAITED	HELP TASK NOT IN USE
16	SHORT WAITED	DC 0 SUSPENDED
12	SHORT WAITED	DC SPACE HANDLER IDLE

VIRTUAL MEMORY LINKS

This section of the dump verifies the virtual memory links in each of the four areas of memory: pseudo-nucleus area, micro slice area, TCB/PCB slice area, and the overlayable area. It analyzes the usage of the overlayable area by listing each data segment and available area by length (in D-words). This section of the printout will be as follows:

ANALYZE MEMORY LINKS	
PSEUDO NUCLEUS	AREA LINKS OK
MICRO SLICE	AREA LINKS OK
TCB/PCB SLICE	AREA LINKS OK
AVAILABLE	: 0642
SEGMENT	: 0048
AVAILABLE	: 9834
OVERLAYABLE	AREA LINKS OK
VM LINKS OK	

PERIPHERAL ASSIGNMENTS AND DESCRIPTORS

This section of the printout will be as follows:

PERIPHERAL ASSIGNMENTS AND DESCRIPTORS	
PORT # 004	DKA POWER OFF
	PACK-ID =
	# OPEN FILE 00000
	DIRECTORY INFORMATION = 0000 0000 0000 0000 0000 0000
	I/O DESCRIPTORS:
	NONE
	DKB SYSTEM RDY
	PACK-ID = CMS8800
	# OPEN FILE 00000
	DIRECTORY INFORMATION = 0020 0004 0024 0008 002F 0071
	I/O DESCRIPTORS:
	NONE
PORT # 006	MTA NOT RDY
	MFID/FID = UNLABELLED
	ASSIGNED TO TASK UNASSIGNED
	REEL #
	I/O DESCRIPTORS:
	NONE
PORT # 002	LPA NOT RDY
	MFID/FID = UNLABELLED
	ASSIGNED TO TASK UNASSIGNED
	I/O DESCRIPTORS:
	NONE
PORT # 003	R8A NOT RDY
	MFID/FID = UNLABELLED
	ASSIGNED TO TASK UNASSIGNED
	I/O DESCRIPTORS:
	NONE
PORT # 001	DCO
PORT # 007	RTC
PORT # 008	SPO RDY
	I/O DESCRIPTORS:
	NONE

Lists each warmstarted device by port number and then lists:

Device Mnemonic Current Status PACK-ID MFIL/FID ASSIGNED TO TASK OPEN FILE REEL DIRECTORY INFORMATION	Code name characterizing a certain device. RDY, NOT RDY, Power Off. File name assigned to device (disk only). File name assigned to device (card readers, tape units, printers only). Mix number of task to which device is currently assigned. For card readers, tape units, printers only. The number of files currently in use on a particular disk. The number assigned to each tape unit; used for multi-reel files. Directory information block - disk only - 12 byte block consisting of 6 fields: (1)non-file directory disk address. (2)length of non-file directory (sectors). (3)file directory disk address. (4)length of file directory (sectors). (5)disk file header disk address. (6)length of disk file header list (sectors).
I/O DESCRIPTORS	I/O descriptors currently in device's queue.

TASK DETAIL TABLE

This section of SYSDUMP consists of one entry per task and contains task history information.

This area of the printout will be as follows:

```

TASK DETAIL TABLE -- DISK COPY
0300 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
0015 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
002A 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
003F 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
0054 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
0069 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
007L 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
0093 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
00AB 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
00B0 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
00D2 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
00E7 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000

S01
0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
0014 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
0028 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
003C 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
0050 AC10 0479 0954 06CA 6000 2810 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
0064 2810 0247 01AF 0000 0000 2810 0264 02E3 0000 0000 2810 0275 0158 0000 0000 0000 0000 0000 0000 0000 0000
0073 2810 025E 00FE 0000 0000 2810 0200 0238 0000 0000 2810 0309 02F6 0000 0000 0000 0000 0000 0000 0000 0000
008C 2810 0282 0352 0000 0000 2810 0000 0000 0000 0000 2810 0671 1139 0000 0000 0000 0000 0000 0000 0000 0000
00A0 2810 0178 020C 0000 0000 2810 0399 0289 0000 0000 2810 0381 016C 0000 0000 0000 0000 0000 0000 0000 0000
00E4 8810 0182 0075 1B94 0001 A810 0185 0125 1C0A 0001 2810 0192 02F8 0000 0000 0000 0000 0000 0000 0000 0000
00C8 2810 03F0 0273 0000 0000 2810 0654 005E 0000 0000 2810 0430 0209 0000 0000 0000 0000 0000 0000 0000 0000
00DC 2810 0452 0270 0000 0000 2810 0461 028F 0000 0000 2810 0470 0131 0000 0000 0000 0000 0000 0000 0000 0000
00FD 2810 018C 018C 0000 0000 2810 01C6 009B 0000 0000 2810 01E2 01A8 0000 0000 0000 0000 0000 0000 0000 0000
0104 2810 01F4 01E8 0000 0000 2810 0200 024C 20A0 0000 2810 01CA 0419 0000 0000 0000 0000 0000 0000 0000 0000
0118 8810 0159 01CE 1889 0001 2810 0613 037F 0000 0000 2810 0628 0318 0000 0000 0000 0000 0000 0000 0000 0000
012C 2810 0638 0383 0000 0000 2810 0602 0599 0000 0000 2810 0668 0068 0000 0000 0000 0000 0000 0000 0000 0000
0140 2810 0503 036F 1030 0000 2810 0527 02C8 0000 0000 2810 0591 037F 0000 0000 0000 0000 0000 0000 0000 0000
0154 2810 0427 0176 0000 0000 2810 065F 020A 0000 0000 2810 0517 0194 0000 0000 0000 0000 0000 0000 0000 0000
0168 2810 0227 0397 0000 0000 2810 0650 00A1 0000 0000 2810 0657 0141 0000 0000 0000 0000 0000 0000 0000 0000
017C 2810 05F0 015C 0000 0000 2810 05F8 0187 0000 0000 2810 023C 01E5 0000 0000 0000 0000 0000 0000 0000 0000
0190 2810 0164 0301 0000 0000 2810 066E 0053 0000 0000 2810 0288 02E0 0000 0000 0000 0000 0000 0000 0000 0000
01A6 2810 02C5 0416 0000 0000 2810 02EA 02A3 0000 0000 2810 02FA 0297 0000 0000 0000 0000 0000 0000 0000 0000
0188 2810 0328 0120 0000 0000 2810 03FF 01E0 0000 0000 2810 040A 01FE 0000 0000 0000 0000 0000 0000 0000 0000
01CC 2810 038A 0280 0000 0000 2810 014C 0236 0000 0000 2810 0336 0298 0000 0000 0000 0000 0000 0000 0000 0000
01E0 2810 0352 0271 0000 0000 2810 0361 0252 0000 0000 2810 036F 0300 0000 0000 0000 0000 0000 0000 0000 0000
01F4 AC10 00C9 0186 0543 6000 3810 05A6 05C8 0000 0000 2810 059C 01B5 0000 0000 0000 0000 0000 0000 0000 0000
0208 3810 010E 0085 0000 0000 3810 0109 00CC 0000 0000 3810 0089 0543 0000 0000 0000 0000 0000 0000 0000 0000
021C 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
0230 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
0244 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
  
```

PRINT TCB

This section of SYSDUMP prints out the contents of the Task Control Block (TCB) of five independent runners and eleven possible user mix numbers. The user task numbers are 1 through 11. The independent runners are the SCL-Loader, the help task, the working set bailiff, the result queue processor, and the data comm space handler (Task 12). This area of the printout will be as follows:

SEGMENT 031
SEGMENT NOT PRESENT

SEGMENT 032
SEGMENT NOT PRESENT

HELP TASK TCB

TCB

POINTERS

0000 0000 103D 1045 1047 1058 0000 00C0 0000 103D 0000 0000 0000

CPA

0000 0000 103D 1045 1047 1058 0000 0000 0000

DST

0000 0010 0000 0037 0000 0510 0000 005A 21FE

STACK

TOS (0 RELATIVE): 0002

0000 0140 8490 4018 4D40 0000 00C0 00C0 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000

SEGMENT 000
SEGMENT NOT PRESENT

SEGMENT 001
SEGMENT NOT PRESENT

WORKING SET BAILLIF TCB

TCB

POINTERS

0000 0000 1067 1067 1068 1077 0000 0000 0000 1067 0000 0000 0000

CPA

0000 0000 1067 1067 1068 1077 0000 00C0 0000

DST

0000

STACK

TOS (0 RELATIVE): 0001

0000 8A79 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000

RESULT QUEUE PROCESSOR
TASK NOT PRESENT

WORK REGISTERS

0000 836C 0888 2184 FFFF FFFF FFFF FFFF FFFF 0000 BD49 0000 1030 0040 0140 BD46 1888
0010 1888 102F 1E44 BD05 BD05 C000 A44D BD58 BFA9 BF72 0068 0002 0000 0000 BE98 0370
0020 102F 0018 000C FC90 0370 2401 2487 0049 0017 4018 FFFF 1804 0000 2810 0200 024C
0030 2070 BFA9 0100 8000 0000 0440 BD05 09EA 3FFF 00FE 0014 FFFF 0005 BFA9 0100 BFF3
0040 0000 BFE8 BD4F 0000 0000 BFA9 0100 BD37 0080 0000 0000 0000 0000 0000 0000 0000
0050 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
0060 0000 0000 0000 0000 0000 0000 00C0 0000 0000 0000 0000 0000 0000 C000 0000 0000 0000
0070 0000 0000 0000 0000 0000 0000 0000 0000 0000 2494 241F 4448 BFCF 8F09 BFA9 0000
0080 BFC2 BFE8 0000 BD49 0000 0000 0000 2500 0076 229E 2256 BFF3 42FA 0057 0002 283E
0090 BD05 0000 0600 8048 0000 0000 00C0 0000 0000 0000 0000 0000 0000 0000 0000 0000
00A0 0000 0000 0000 0000 0000 0000 BF83 4229 0022 0000 0000 0000 0000 0000 0000 0000
00B0 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
00C0 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
00D0 0000 0000 0000 0000 0000 0000 00C0 0000 0000 0000 0000 0000 0000 0000 0000 0000
00E0 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
00F0 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000

If a task is present, the Task Control Block is broken out as follows:

<p>Pointers</p> <p>PSN/ISN DST Base DST Limit TOS Stack Limit MCH TOS Recover</p> <p>MCH Active Verb TCB Flags MRA Base COMM Result A COMM Result B S-Start PCA</p> <p>IPA</p> <p>CPA</p> <p>DST</p> <p>Stack</p> <p>Data Segments</p>	<p>The first 12 entries of the TCB:</p> <p>The entry numbers in the SDT of the Program Control Block and the Interpreter. Absolute address of base of Data Segment Table. Address of the first word after DST. Address of the next available entry of data stack. Address of first word after stack. The difference between the present TOS and the base of the TCB. Use for error recovery in the Master Communicate Handler. Relative address of the CPA from the base of the TCB. Flags used by the operating system to interface between system modules. Address of base of Message Reference Area (used by data comm tasks). Result of an I/O communicate returned by some OS module.</p> <p>Program Count Address. The Interpreter's offset into the current program segment.</p> <p>The Interpreter Preset Area is a work area where the S-Interpreter will store relative addresses before relinquishing control to the operating system.</p> <p>The Communicate Parameter Area contains all the parameters and variables required for a particular communicate S-Operation.</p> <p>Data Segment Descriptors listed in the order of increasing segment number.</p> <p>Area in TCB used by the OS for control and storage of parameters, data, relative returns, etc.</p> <p>Each data segment which is currently present in memory listed in order of increasing segment number. A data segment which is an FIB is listed as such and if it is open SYSDUMP also lists certain pertinent information from its FIB.</p>
--	--

PRINT SLICES

This section of the dump prints the disk open and close slices, the allocate, deallocate slice and the reconstructor slice if they are present in memory.

DATA COMM AREA ANALYSIS (ONLY IF RELEVANT)

<p>Registers</p> <p>DC LOCK</p> <p>BUFFER SIZE</p> <p>LOGICAL LISTS</p> <p>LINE STATION TERMINAL ADDRESS</p> <p>SUBNET Q TABLE</p> <p>DC XREGS</p> <p>AVAILABLE BUFFER POOL</p> <p>HEAD TAIL COUNT</p> <p>STATUS</p> <p>RESERVE BUFFER POOL</p> <p>HEAD TAIL COUNT LIMIT</p> <p>XTENT BASE</p>	<p>Contains the lockout-ID of the processor which is currently in control of the queue of available buffers.</p> <p>Contains the text capacity of a DC buffer. It equals the DC buffer size minus two specified in one's complement form.</p> <p>Address of list of line table addresses. Address of list of station table addresses. Address of list of terminal table addresses.</p> <p>Address of the subnet queue table or all ones if no files were declared in the NDL program.</p> <p>Address of the base of the table known as "DC Extended Registers".</p> <p>The address of the first buffer in the queue of available buffers. The address of the final buffer in the queue of available buffers. The number of data comm buffers which are currently available for use. Any buffer that is currently being used as part of a message is not counted. Contains run time information as to whether or not a particular task is being waited because of a deficiency of available buffers.</p> <p>The address of the first buffer being held in the reserve pool. The address of the final buffer being held in the reserve pool. The number of buffers currently being held in the reserve pool. The number of buffers needed in the reserve pool before the space can be released to the operating system. The highest address that can be written a DC buffer in the reserve area. The lowest address that can be within a DC buffer in the reserve area.</p>
--	---

PMLC 0 REQUEST HEAD TAIL RESULT HEAD TAIL PMLC 1 REQUEST HEAD TAIL RESULT HEAD TAIL	The address of the next message to be processed by PMLC-0. The address of the final message that is currently waiting to be processed by PMLC-0. The address of the next message that has been processed by the PMLC-0 firmware but has not yet been handled by the PMLC interrupt processor. The address of the final message that has been processed by the PMLC-0 firmware but has not yet been handled by the PMLC interrupt processor. See PMLC 0 REQUEST, HEAD. See PMLC 0 REQUEST, TAIL. See PMLC 0 RESULT, HEAD. See PMLC 0 RESULT, TAIL.
--	--

DC EXTENDED REGS

Contains implementation dependent DC registers and pointers that are not appropriate in any of the NDL tables.

LINE TABLES

Contains information about the line as described in the NDL program.

STATION TABLES

Contains information about the station as described in the NDL program.

TERMINAL TABLES

Contains information about the terminal as described in the NDL program.

SUBNET QUEUE TABLE

Contains implementation dependent registers and points for each subnet queue (file) defined in the NDL program.

PMLC 0 Request Buffers PMLC 0 Result Buffers PMLC 1 Request Buffers PMLC 1 Result Buffers	Contains pointers to messages that have not been processed by PMLC 0. Contains pointers to messages that have been processed by PMLC 0 and are being forwarded to the data comm module for further processing. See PMLC 0 Request Buffers. See PMLC 0 Result Buffers.
--	--

AVAILABLE BUFFER POOL

Contains pointers to DC buffers that are not currently in use.

RESERVE BUFFER POOL

Contains pointers to DC buffers that are not currently in use and reside in an area that is about to be returned to the OS overlayable data area.

DC STATE SPACE

Contains all of the data comm tables and buffers that reside in main memory as well as the NDL S-OPs.

MPLII DUMP ANALYZER (MPL2DUMP)

The MPLII Dump Analyzer is a debugging aid for the programmer. When an MPL program is DP-ed, a dump file is created on disk. MPL2DUMP examines this file and arranges it into an easily readable format.

OPERATING INSTRUCTIONS

MPL2DUMP may be executed any time the system is operating by entering via the system console:

```
MPL2DUMP filename HEX
```

“filename” is the name of the dump file created when the MPL program was DP'ed. It will reside on the same disk as the object code file.

HEX is an option that, if entered, will produce an unformatted dump. Default is a formatted dump.

ERROR MESSAGES

If an error is detected in the execute command, MPL2DUMP will display the message:

```
*** INVALID UTILITY PARAMETER ***
```

Check the execute statement just entered as described in the procedure below:

1. Check that the file-id of the dump file was entered. If it was not entered, re-enter the execute statement and be sure to include the file-id.
2. If the file-id was entered, count the number of characters in the file-id. There should be 12 characters or less. If the file-id was greater than 12 characters, re-enter the execute statement using the correct file-id.
3. If the file-id was entered correctly, check if the dump file is on the disk. If the dump file is not on the disk, then the disk containing the dump file must be installed in a disk drive.
4. If the pack-id was entered, check that it does not contain more than 7 characters. If the pack-id was entered incorrectly, re-enter the execute statement with the correct pack-id.

If an error is detected in the dump file, MPL2DUMP will display the message

```
*** ERROR DETECTED IN DUMP FILE ***
```

If an unformatted dump has not been produced, MPL2DUMP will print one and terminate automatically.

EXPLANATION OF FORMATTED DUMPS

This section explains and gives examples of a formatted dump analysis listing in the actual order listed.

NOTE

Whenever both the hexadecimal (hex) and decimal values are given, they will appear in the form:

HHHH/DDDDD

where

HHHH is the hex and DDDDD is the decimal equivalent.

The following paragraphs explain the individual sections of the analysis.

HEADING

The format and content of the first page heading will be as follows:

```
MPL2 DUMP ANALYZER 04/29/77  
PROGRAM ID        CHS8800/KA  
DMFILE ID         0000000/DMFIL11  
REASON FOR DUMP:  [000] USER REQUESTED
```

The actual data, program identification, dumpfile identification, and reason for the dump will vary accordingly.

S-REGISTERS

The format and content of the S-Register section will be as follows:

S-REGISTERS -----				
PSN	0003/00003	LVL	0002/00002	
SPN	0000/00000	REG1	0001/00001	
PCA	0133/00307	REG2	0000/00000	
LSA	0001/00001	CARRY	0013/00019	
NLD	0007/00007	PCC	0002/00002	
STA	057A/01402	NMR	0000/00000	
TOS	0000/00013			
MODE	PROCESS			

The register name is listed and the data it stores is listed next to it. The following identifies the data.

Register Name	Data Stored in Register
PSN	Segment number of the currently active procedure.
SPN	Procedure number of the currently active procedure.
PCA	Address of the S-instruction currently being executed.
LSA	Size of the literal pool for the currently active procedure.
NLD	Number of local descriptors in the currently active procedure.
STA	Address of the next available byte in the data stack (segment 0).
TOS	Address of the next available byte in the control stack.
MODE	Mode of the MPLII Virtual Machine: PROCESS, REMAP, or DECLARE.
LVL	The lexical (lex) level of the currently active procedure.
REG1	The lex level of the most referenced region.
REG2	The lex level of the next most referenced region.
CARRY	Interpreter arithmetic work register used by certain S-OPs (DIVIDE, MOD, etc).
PCC	Number of the currently called procedure.
NMR	Number of the next available message reference.

DECLARE (DECL) REGISTERS

The format and content of the Declare Registers section will be as follows:

DECL REGISTERS -----			
NDA	057A/01402	SOL	0233/00563
SEGN	0000/00000	EOL	0273/00627

The following table lists the registers and identifies their contents.

Register Name	Data Stored in Register
NDA	Next Descriptor Address. The offset into segment 0 of the next descriptor to be declared.
SEGN	Segment number of the data currently being declared.
SOL	Start Of Last. The offset into segment number SEGN of the start of the last descriptor's data area.
EOL	End Of Last. The offset into segment number SEGN of the end of the last descriptor's data area.

DISPLAY

The display section contains the lex level and corresponding offset into segment 0 of the base of the descriptors for each lex level's most recent procedure invocation. The format and content of the display section will be as follows:

DISPLAY	

LEX LEVEL	DATA STACK PTR
00	0000/00000
01	0355/00853
02	0550/01373
03	0000/00000
04	0000/00000
05	0000/00000
06	0000/00000
07	0000/00000
08	0000/00000
09	0000/00000
10	0000/00000
11	0000/00000
12	0000/00000
13	0000/00000
14	0000/00000
15	0000/00000

MISCELLANEOUS (MISC)

The format and content of the MISC section will be as follows:

MISC	

FETCH VALUE	000000
OPCODE	0073
TEMPS	0002 0383 0003 03FC 3631 FFFF 0002 00CA FF00 13F8 20CE 2710
CPA	8402 1010 0010 0000 C000 0000 0000 0000

MISCELLANEOUS (MISC) (Continued)

The table below lists the miscellaneous registers and identifies their contents.

Register Name	Data Stored in Register
FETCH VALUE	Last communicate response.
OPCODE	The opcode of the current instruction.
TEMPS	Internal temporary registers used by the interpreter.
CPA	Communicate Parameter Area. Last communicate issued to the virtual machine.

TASK CONTROL BLOCK (TCB)

The format and contents of the TCB Registers section will be as follows:

TCB REGISTERS	011E/00286 BYTES

0000/00000	6C10 2219 2241 2314 234A 00C5 0079 0151 2219 0000 0000 0133 0003 0000 0000 0000
0020/00032	0000 0000 0000 0000 C000 0C00 0000 0000 0000 0000 FFFF 0282 FFFF 270C 0000 0000
0040/00064	8001 0000 0000 2710 20CE 0006 0000 03FC 3031 0000 0000 0000 0000 0073 06CE 0E37
0060/00096	0E74 0000 067C 0949 C010 06CA 0000 0000 0013 0000 0355 0550 0000 0000 0000 0000
0080/00128	0000 0000 0000 0000 C000 0000 0000 0000 0000 0001 0002 0007 0002 0001 0000 0000
00A0/00160	057A 0273 0233 0000 057A 0002 0383 0003 03FC 3631 FFFF 0002 00CA FF00 13F8 20CE
00CC/00192	2710 0000 0000 0000 C000 0C00 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
00EC/00224	0000 0000 0000 0000 C000 0000 0000 0000 0000 8402 1010 0010 0000 0000 0000 0000
0100/00256	0000 0000 00C4 0000 C000 0000 0000 0000 BFA9 015A D501 BF80 0000 0F27 15EB FF72

Next to the title is the length given in bytes. Below the section title, the relative byte address of the first register of each line is listed. Next to each address is an unedited hexadecimal printout of the contents of those registers. The contents of the MISC section are included in this section.

MESSAGE (MSG) REFERENCE AREA

The format and contents of this section will be as follows:

```

MSG REFERENCE AREA 0000/00000 BYTES
-----

```

This section contains an unedited hexadecimal printout of the program's message reference area.

DATA SEGMENT TABLE

The format and contents of this section will be as follows:

```

DATA SEGMENT TABLE 0050/00080 BYTES
-----
0000/00000      8210 0000 1388 234A 8510 0ED5 00C9 6879 9200 0000 00A3 3602 1210 0008 0000 0000
0020/00032      1200 0009 0000 0000 8710 0ED8 0126 408A 8310 0F10 01FE 3031 8310 0F24 0020 6C85
0040/00064      8710 0F25 0020 6C43 8710 0F26 0020 6C64

```

CONTROL STACK

The format and contents of this section will be as follows:

```

CONTROL STACK 0212/00530 BYTES
-----
0000/00000      0000 0000 0000 0000 C220 0C00 581B 0000 0206 0306 1001 1C00 0000 0328 89BA 8A6A
0020/00032      007A 2200 1EFD 10C4 0010 939F 9170 1EFD 10C4 0010 939F 9170 0000 0001 FFFF 0000
0040/00064      203A 6C85 0002 0042 C001 0068 600C 0000 0209 0002 0007 0215 898A 8A6A 0A00 0000
0060/00096      0000 0000 0C00 0000 C000 0C00 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
0080/00128      0000 0000 0000 0000 0000 0C00 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
00AC/00160      0000 0000 0000 0000 C000 0C00 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
00CC/00192      0000 0000 0000 0000 C000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
00EC/00224      0000 0000 0000 0000 0000 0C00 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
0100/00256      0000 0000 0C00 0000 C000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
0120/00288      0000 0000 0000 0000 C000 0C00 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
0140/00320      0000 0000 0000 0000 C000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
0160/00352      0000 0000 0000 0000 C000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
0180/00384      0000 0000 0000 0000 C000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 2609
01A0/00416      7F00 0000 0428 101C GA00 2000 0000 BFC2 0080 000A 68FF 4C50 4120 0000 0000 0000
01C0/00448      0020 2000 0000 8FCF C042 0015 69FF 5238 4120 0000 0000 0000 0020 2000 3AF8 7F00
01E0/00480      0000 0358 3AF8 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
0200/00512      0000 0000 0000 0000 C000 0000 0000 0000 0000

```

This section of the listing is an unedited printout of the control stack.

DATA SEGMENTS

The format and contents of this section will be listed as follows:

```

DATA SEGMENT 00      2710/10000 BYTES
-----

SEGMENT ATTRIBUTES:
PRESENT
LOCKED
READ/WRITE

0030/00000  0101 0178 0101 01BB 0184 01FB 0178 01FB 010C 01FB 0101 01FB 0107 0207 0002 020E
0020/00332  0106 0210 0107 02AF C101 02AF 010C 0286 0101 0286 0107 02C2 0103 02C9 0101 02C9
0040/00064  0002 02CA 0101 02CA C284 02CA 0101 02CB 0120 02CC 0281 02CC FF00 0000 FF00 0000
0060/00096  FF00 0000 FF00 0000 FF00 0001 FF00 0000 FF00 0000 FF00 01F2 FF00 010A FF00 0000
0080/00128  FF00 0000 FF00 0000 FF00 00C6 FF00 0000 FF00 0000 FF00 00B0 FF00 0040 FF00 0000
0100/00160  FF00 0000 FF00 00AE FF00 0000 FF00 01F8 FF00 024C FF00 03FC FF00 0000 FF00 002F
00CC/00192  FF00 0024 FF00 002F FF00 0024 FF00 024C FF00 03FC FF00 036C FF00 0000 FF00 0000
00E0/00224  FF00 32C0 FF00 0000 FF00 0000 FF00 0000 FF00 0000 FF00 0000 FF00 0000 FF00 0000
0100/00256  FF00 0000 FF00 0000 FF00 0000 FF00 0000 FF00 0000 FF00 0001 FF00 0001 FF00 0000
0120/00288  FF00 FF88 FF00 0000 C107 02EC 0101 02EC 013C 02F3 0120 02F3 0117 02F3 0101 02F3
0140/00320  0106 032F 010E 0335 C501 0000 0550 0000 0106 0343 010C 0349 1506 0000 190C 0000
0160/00352  0000 0000 0000 0000 C000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
0180/00384  C6C7 C8C9 D1D2 D3D4 D5D6 D7D8 D9E2 E3E4 E5E6 E7E8 E9F0 F1F2 F3F4 F5F6 F7F8 F940
01A0/00416  4A48 4C4D 4E50 5A5B 5C5D 5E60 6168 6C6E 6F7A 787C 7E7F 7F00 0000 0041 4243 4445
01C0/00448  4647 4849 4A48 4C4D 4E4F 5051 5253 5455 5657 5859 5A30 3132 3334 3536 3738 3920
01E0/00480  582E 3C28 2826 5024 2A29 3820 2F2C 253E 3F3A 2340 3022 0000 0000 0020 2020 2020
0200/00512  3531 3631 2020 2020 2020 2020 2020 2020 2033 3220 2020 2020 4153 5349 474E 4544
0220/00544  2020 2020 2020 2041 4045 4E44 2020 2020 2020 2020 2020 2020 2020 2020 2020 2020
0240/00576  2020 2020 2020 2020 2020 2020 2020 2020 2020 2020 2020 2020 2020 2020 2020 2020
0260/00608  2020 2020 2020 2020 2020 2020 2020 2020 2000 0000 0100 0100 0000 0100 0100
0280/00640  0000 0100 0100 0000 C100 0100 0000 0100 0100 0000 0100 0100 0000 0100 0100 0000
02A0/00672  0100 0100 0000 0100 C100 00C0 0000 0030 3030 3030 3030 2020 2020 2020 2020 2020
02C0/00704  2020 4340 5342 3830 3000 0010 0000 0000 FF56 FFFB FFFF FFFF FFFE 0000 0000
02E0/00736  0000 0030 0000 0000 0000 0000 3030 3030 3030 3000 0000 0000 0000 0000 0000 0000
0300/00768  0000 0000 0000 0000 C000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
0320/00800  0000 0030 0000 0000 C000 0000 0000 0000 00F1 F3F5 F7F9 F054 494F 4454 4E20 2020 2020
0340/00832  2020 2030 0000 0000 C020 C020 2020 2020 2020 2020 2015 0600 0019 0C00 0014 0200
0360/00864  0014 0200 0214 0200 0416 8400 0419 0C00 0019 0100 0001 7801 FBFF 0013 F8FF 0000
0380/00896  31FF 0014 29FF 0000 20FF 0014 49FF 0000 2601 7803 C501 3C03 C501 0003 C501 0703
03A0/00928  E001 3C04 0101 0604 C101 0E04 0F01 0404 2401 7804 3D01 7804 B501 0C05 20FF 0000
03C0/00960  01FF 0000 2746 5249 2032 3920 4150 5220 3737 2020 4449 5348 2041 5245 4120 5553
03E0/00992  4147 4520 4F46 2044 502E 4944 2043 4053 4238 3030 2020 5345 5249 414C 204E 4F2E
0400/01024  2031 3335 3739 3020 204F 574E 4552 2054 494F 4454 4E20 2020 2020 2020 2020 2050
0420/01056  4147 4520 2020 2031 2E20 2020 2020 2020 2020 2020 2020 2020 2020 2020 2041 5245
0440/01088  4120 4144 4452 4553 5320 2020 4152 4541 204C 454E 4754 4820 2020 2020 2053 5441 5455
0460/01120  5320 2020 2020 2020 2046 494C 4520 4E41 4045 2020 2020 2020 2020 2020 2020 2020
0480/01152  2020 2020 2020 2020 2020 2020 2020 2020 2020 2020 2020 2020 2020 2020 2020 2020

```

This section of the listing contains information about each of the program's data segments. The segment's size and attributes are obtained from the data segment table. Following this information, is an unedited hex dump of that segment.

When a data segment is assigned the attributes ABSENT and FIB, the virtual machine had not yet formed that segment at the time of the dump. A sample of this is shown in the following:

```

DATA SEGMENT 02      0146/00326 BYTES
-----

SEGMENT ATTRIBUTES:
PRESENT
LOCKED
READ/WRITE
FIB

```

Data segment 00 contains all the data descriptors for the task. It is the data stack. The information contained in segment 00 is further analyzed during the control stack analysis.

FILE ANALYSIS

This section of the listing contains information about the task's files. A sample analysis is shown in the following for both open and closed files.

FILE ANALYSIS	
FILE	(OPEN)
FIB SEGMENT	0002/00002
FPB SEGMENT	0007/00007
FILE STATE:	
WRITE	
OUTPUT	
FILE TYPE	0000 DATA
DEVICE KIND	UNKNOWN
RECORD LENGTH	0000/00000
BUFFER LENGTH	0000/00000
# BUFFERS	0000/00000
MAX FILE SIZE	00000000
MYUSE	ILLEGAL
OTHERUSE	LOCK
ACCESSMODE	ILLEGAL
FILE	(CLOSED)
FIB SEGMENT	0003/00003
FPB SEGMENT	0008/00008
FILE TYPE	0000 DATA
DEVICE KIND	UNKNOWN
RECORD LENGTH	0000/00000
BUFFER LENGTH	0000/00000
# BUFFERS	0000/00000
MAX FILE SIZE	00000000
MYUSE	ILLEGAL
OTHERUSE	LOCK
ACCESSMODE	ILLEGAL

CONTROL STACK

This portion of the dump decodes the information in the stack and lists the data descriptors and data values of all the task's currently active variables. The analysis begins with the oldest procedure and progresses level by level to the most recently called procedure. A sample analysis is shown in the following.

CONTROL STACK ANALYSIS

NEXT LEVEL

PSN	0000/00000	REG1	0000/00000
SPN	0000/00000	REG2	0000/00000
PCA	0220/00544	NLD	0058/00088
LVL	0000/00000	LSA	001B/00027

ADDRESS	DESCRIPTOR	OCCUR	DATA TYPE	SEG ORIGIN	DATA
0003/0C000	01010178	000	CHARACTER(001)	00 0178	C1
0004/00004	01010188	001	CHARACTER(001)	00 0188	41
0008/0C008	01B401F8	002	CHARACTER(180)	00 01F8	20202020203531363120202020202020 20202020203332202020202020415353 49474E4544202020202020204140454E 44202020202020202020202020202020 20202020202020202020202020202020 20202020202020202020202020202020 20202020202020202020202020202020

REGISTERS

The registers and their functions are listed in the following table.

Register Name	Data Stored in Register
PSN SPN	The segment number of the currently active procedure. The procedure number of the currently active procedure.
PCA	The address of the S-instruction currently being executed.
LVL	The lex level of the currently active procedure.
REG1	The lex level of the most referenced region.
REG2	The lex level of the most referenced region.
NLD	The number of local descriptors in the procedure.
LSA	The size of the literal pool for the currently active procedure.

DESCRIPTOR INFORMATION

The descriptor information is formatted into seven columns with the following headings:

Heading	Data in Column
ADDRESS	The byte offset into segment 00 of the start of the data description.
DESCRIPTOR	The actual value of the four-byte descriptor.
OCCUR	The occurrence number of the data descriptor.
DATA TYPE	The data can be of five types: FIXED, Bit (00=00), MSG REFERENCE, SELF RELATIVE, and CHARACTER (000). The BIT data type is followed by parentheses. The data enclosed is: bit position=bit length. The CHARACTER data type is followed by its length enclosed in parentheses.
SEG	The segment number of the data.
ORIGIN	The byte offset in segment <SEG> of the data.
DATA	The value of the data stored at that address.

COBOL DUMP ANALYZER (COBOLDUMP)

The COBOL Dump Analyzer is a debugging aid for the programmer. When a COBOL program is DP-ed, a dump file is created on disk. COBOLDUMP examines this file and arranges it into an easily readable format.

OPERATING INSTRUCTIONS

COBOLDUMP may be executed any time the system is operating by entering via the system console:

```
COBOLDUMP filename HEX
```

“filename” is the name of the dump file created when the COBOL program was DP-ed. It will reside on the same disk as the program code file.

HEX is an option that, if entered, will produce an unformatted dump. Default is a formatted dump.

After a formatted dump, COBOLDUMP will go to EOJ. After an unformatted dump, the program will display the message FORMATTED DUMP?. If a formatted dump is desired, the alphabetic character Y is entered via the AX command. COBOLDUMP will terminate automatically after the formatted dump. If the formatted dump is not desired, the alphabetic character N causes COBOLDUMP to go to EOJ.

ERROR MESSAGES

The same error messages, causes, and corrections apply to COBOLDUMP as was defined for MPL2DUMP.

EXPLANATION OF FORMATTED DUMPS

This section explains and gives examples of a formatted dump analysis listing in the actual order listed.

NOTE

Whenever both the hexadecimal (hex) and decimal values are given, they will appear in the form:

HHHH/DDDDD

where

HHHH is the hex value and DDDDD is the decimal equivalent.

The following paragraphs explain the individual sections of the analysis.

HEADING

The format and content of the first page heading section of the printout will be as follows:

```
                                COBOL DUMP/ANALYZER
                                -----

DATE           : 04/29/77
DUMPFIL NAME   : 000000/DHFIL01
CODEFIL NAME   : CHS201/TEAM-STATS
REASON FOR DUMP :      0 UNKNOWN
```

The actual data, program identification, dumpfile identification, and reason for the dump will vary accordingly.

PROGRAM PARAMETER BLOCK (PPB) OF CODE FILE

This section of the printout defines the structure within the code file and provides identifying information concerning the program.

The format and content of the PPB section of the printout will be as follows:

```
-----
PPB OF CODE FILE
-----

IMPLEMENTATION NUMBER : 00
PROGRAM NAME          : TEAM-STATS
S.LANGUAGE NAME       : COBOL
INTERPRETER PACK      : 0000000
INTERPRETER NAME      : COBOLINT
COMPILER NAME         : SL9/COBOL
COMPILATION DATE      : 03/11/77
PRIORITY CLASS        : 0400
INIT SEGMENT NUMBER   : FF
INTERP ENTRY SEGMENT : 00
INTERP ENTRY DISP.    : 0017
PST LENGTH            : 0006
PST LOCATION          : 0007
DST LENGTH            : 003C
DST LOCATION          : 0008
STACK LENGTH          : 003C
COP TABLE LENGTH     : 0234
COP TABLE ADDRESS    : 0003
```

The program parameter is listed and the field content it stores is listed next to it. The table below identifies the field content.

Program Parameter	Field Contents
IMPLEMENTATION NUMBER	Level number of the implementation.
PROGRAM NAME	Name of user's program.
S.LANGUAGE NAME	Name of language created by compiler.
INTERPRETER PACK	Pack on which the interpreter resides.
INTERPRETER NAME	Name of interpreter.
COMPILATION DATE	Date of compilation.
PRIORITY CLASS	Priority class of program.
INIT SEGMENT NUMBER	Initiating message segment number.
INTERP ENTRY SEGMENT	Segment in which execution of the program begins.
INTERP ENTRY DISP.	Byte displacement into interpreter entry segment at which execution begins.
PST LENGTH	Length of program segment table.
PST LOCATION	Location of program segment table in CODE-FILE.
DST LENGTH	Length of data segment table.
DST LOCATION	Location of data segment table in CODE.FILE.
STACK LENGTH	Length of control stack.
COP TABLE LENGTH	Length of current operation table.
COP TABLE ADDRESS	Location of current operation table in CODE.FILE.

TASK CONTROL BLOCK (TCB) PRESET AREA

The format and content of the TCB section of the printout will be as follows:

```

-----
TCB PRESET AREA
-----

PSN           : 0068
ISN           : 0013
DST BASE     : 2271
DST LIMIT    : 2299
TOP OF STACK : 22C2
STACK LIMIT  : 22F8
MCH TOS RECOVER : 00B8
MCH ACTIVE VERB : 007A
TCB-FLAGS   : 0051
MRA-BASE    : 2271
COMM RESULT A : 0040
COMM RESULT B : 0000
S-START PCA  : 0034 ( 52)
S-START PSN  : 0000 ( 0)
LAST COMM RESP : 000000
CURRENT OP   : 002F

OVERFLOW FLAG : 0000 (RESET)
LINE COUNT    : 0076 ( 118)
A REGISTER    : 0000 0000 0C00 0000
B REGISTER    : 0000 0000 0C00 0000
C REGISTER    : 0000 0000 0C00 0000

00000 0000 0000 0000 0C00 0000 0000 0000 0000 0000 0000 0000 0000 0000 2070 0452 FFFF
00764 0000 0000 0000 0C00 8000 0019 0420 2096 8000 0014 042A 2091 0000 0000 0000 0000
00128 002F 11AF 0000 18CF 0000 067C 11C9 001A 101F 0000 0076 0000 0000 0000 0000 0000
00192 0001 0000 8000 0C00 0002 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
00256 0000 0000 0000 0C00 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
00320 0000 0000 0002 0C00 0048 0002 0002 0000 0000 0000 0000 0000 0000 0000 0000 0000
00384 1021 0000 0000 0C00 0000 0000 0000 0000 0000 0000 0000 0000 0000 0103 0000 0040
00448 0000 0000 0000 0000 0000 0088 0000 0000 0000 0000 0000 0000 0000 0000 0000 0040
00512 0116 FF72

```

The preset program parameter or register is listed and the field content it stores is listed next to it.

Preset Parameter or Register	Field Contents
PSN/ISN	The first 12 entries of the task control block, reflecting the entry numbers in the slice descriptor table of the program control block and the interpreter.
DST BASE	Absolute address of the base of the data segment table.
DST LIMIT	Address of last entry of DST + 1.
TOP OF STACK	Absolute address of base of partial interpreter stack.
STACK LIMIT	Address of last entry of stack + 1.
MCH TOS RECOVER	Difference between the present top of stack and the base of the task control block. Used for error recovery in the master communicate handler.
MCH ACTIVE VERB	Relative address of the current program address from the base of the task control block.
TCB.FLAGS	Flags used by the operating system to interface between system modules.
MRA.BASE	Address of base of message reference area (used by data comm).
COMM RESULT A	Result of an I/O communicate.
COMM RESULT B	Result returned by an operating system module.
S-START PCA	Program count address. The interpreter's offset into the current program segment.
S-START PSN	The interpreter segment offset into the program's code file.
LAST COMM RESP	Last Communicate response.
CURRENT OP	The operation currently being executed.
OVERFLOW FLAG	Indicates if overflow is present.
LINE COUNT	Line of program currently being executed.
A REGISTER	Contents of the registers that the interpreter uses for numeric operations.
B REGISTER	
C REGISTER	

DATA SEGMENT TABLE

The format and content of the data segment table section of the printout will be as follows:

----- DATA SEGMENT TABLE -----			
0510	30F6	0091	22F5
0510	30F8	0015	22F5
8710	30F9	002F	2758
1200	0032	0000	0000
0510	30FA	0018	22F5
0510	30FB	002F	22F5
1200	0035	0000	0000
0510	30FC	0042	22F5
0510	30FD	05BF	22F5
0510	310E	003C	22F5

This section contains an unedited hexadecimal printout of all eight-byte table entries.

CONTROL STACK

The format and content of the control stack section of the printout will be as follows:

----- CONTROL STACK -----															
00300	0164	007A	0114	8490	0328	89BA	8A6A	8A6A	4152	2020	2020	2020	2020	2020	2020
00364	2020	4441	5441	2D49	4E20	1000	0009	0123	0002	0C01	0000	0000	0000	0000	0000
00128	0000	0000	0000	0C90	0000	0000	0000	0000	0428	131C	0000	0000	0000	0000	0000
00192	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
00256	C000	0000	0000	0C00	0000	0000	0000	0000	0000	0000	0000	0000	4305	0000	0101
00320	0000	0000	0100	0037	00B4	000A	C3FF	0000	0000	00B4	BFA9	002F	0038	0037	0007

The control stack dump is an unedited hexadecimal printout of the control stack contents. The first column of the printout gives the decimal values of the control stack address of each first hexadecimal character in that row. Each control stack entry consists of three words, the last entry being at the head of the control stack.

The format of each entry is:

First Word, First Byte First Word, Second Byte Second Word Third Word	K. - Value sometimes used by the interpreter to determine whether or not to use the stack entry to obtain the address of the next S-instruction to be executed. Segment No. - The number of the code segment. Digit displacement of the next sequential instruction relative to the segment base. Line Count - Line of program being executed.
--	---

DATA SEGMENTS

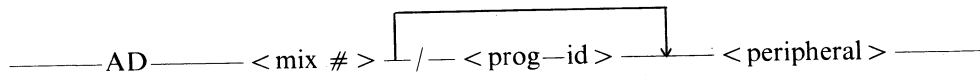
The format and contents of this section will be listed as shown in the following:

DATA SEGMENT 0000/00000	
00300 00064	0000 0000
00128 00192 00256 00320 00384 00448 00512 00576	0000 0000 0000 0C00 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0C00 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0C00 0000 0000 2100 8020 0054 0080 2000 5401 8023 0080 2154 0280 0000 0000 0000 0C00 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 5052 4F47 5241 4020 4142 4F52 5445 4420 2020 5553 4520 5052 4F43 4544 5552 4520 4049 5353 494E 474E 4F20 5459 5045 2032 4E4F 2054 5950 4520 3150 524F 4752 4140 2041 424F 5254 4544 2020 204E 4F20 5354 4F50 2052 554E 2045 4E43 4F55 4E54 4552 4544
00300 00064	0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0C00 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
00128 00192 00256 00320 00384 00448 00512 00576	0000 0000 0000 0C00 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0C00 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0C00 0000 0000 2100 8020 0054 0080 2000 5401 8023 0080 2154 0280 0000 0000 0000 0C00 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 5052 4F47 5241 4020 4142 4F52 5445 4420 2020 5553 4520 5052 4F43 4544 5552 4520 4049 5353 494E 474E 4F20 5459 5045 2032 4E4F 2054 5950 4520 3150 524F 4752 4140 2041 424F 5254 4544 2020 204E 4F20 5354 4F50 2052 554E 2045 4E43 4F55 4E54 4552 4544
DATA SEGMENT 0001/00001	
00300 00064	0000 0000 0000 0C00 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0C00 0000
DATA SEGMENT 0002/00002	
00000 00364 00128	0030 3030 3030 3C30 4441 5441 2049 4E20 2020 2020 2030 3031 0000 0000 1104 0000 0030 0030 0000 0002 0000 3108 3031 0000 3737 3131 3900 0000 0000 0000 3030 3100 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000

APPENDIX A

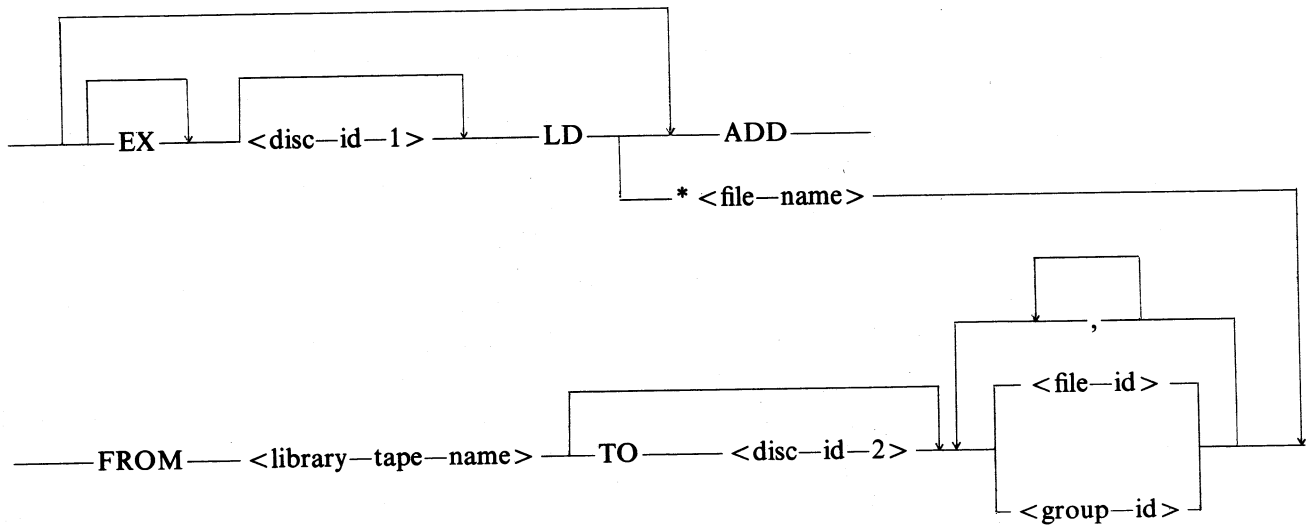
AD

AD (Assign Peripheral)



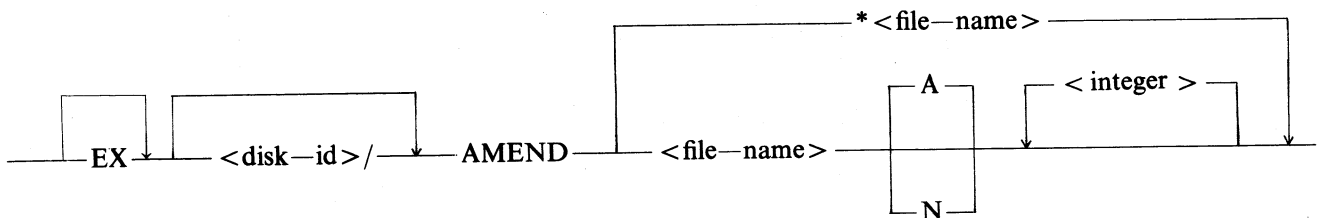
ADD

ADD (Add Files from Library Tape)



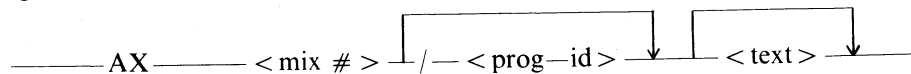
AMEND

AMEND (Disk file amending)



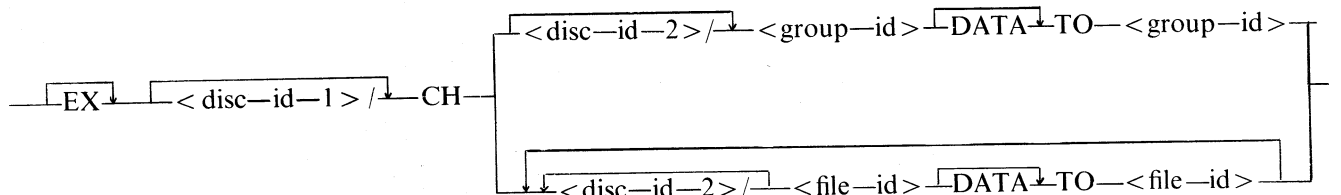
AX

AX (Accept a message for a task)



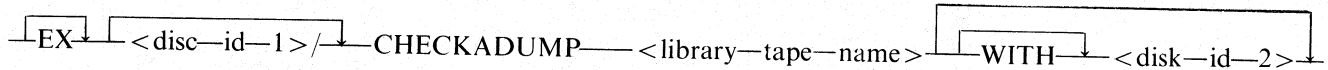
CH

CH (Change <file-identifiers>)



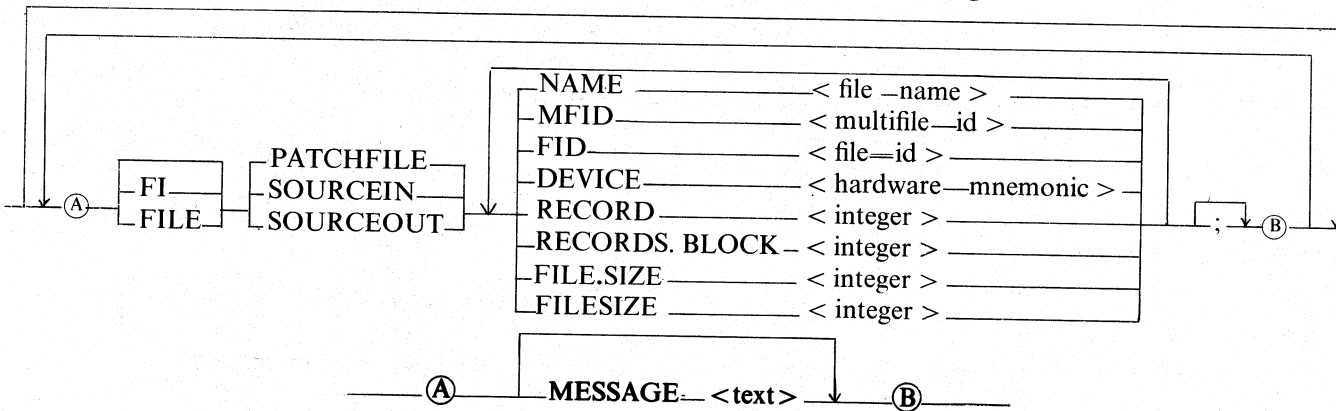
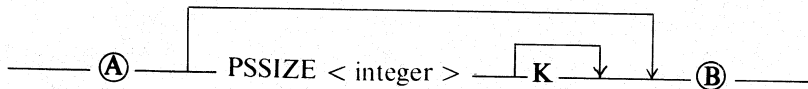
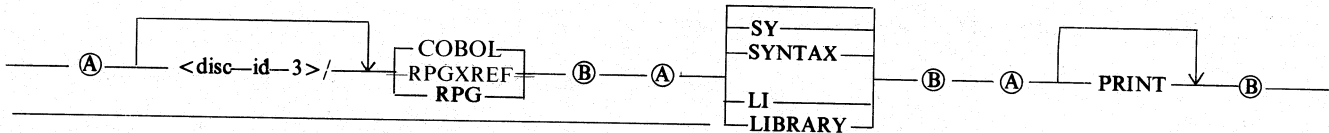
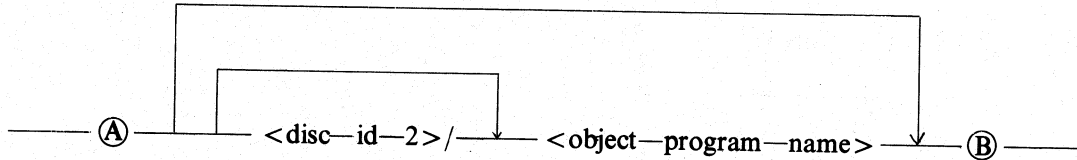
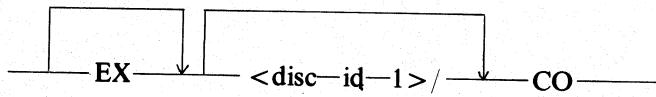
CHECKADUMP

CHECKADUMP (Compare Library Tape with Disk)

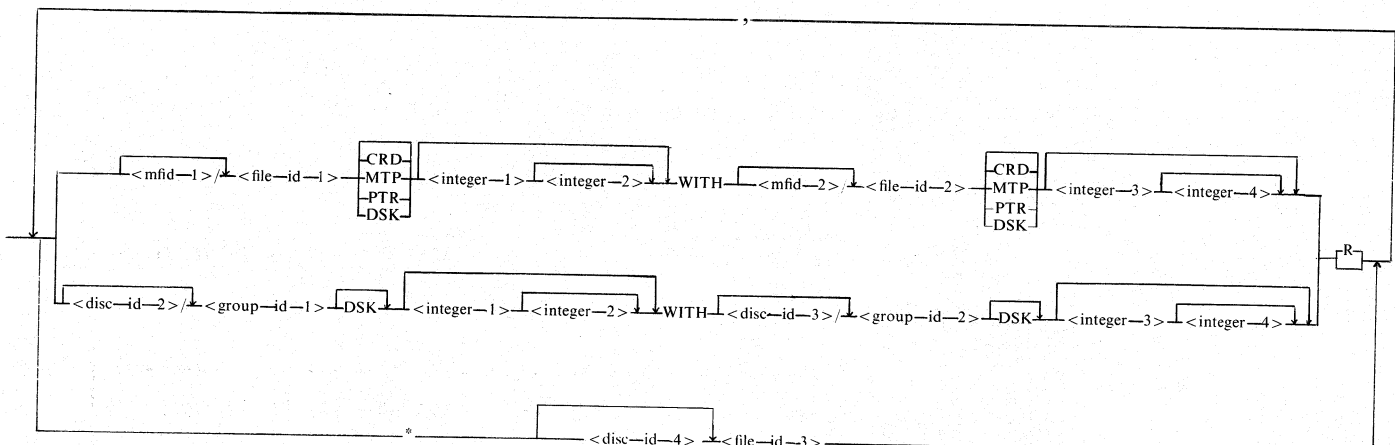
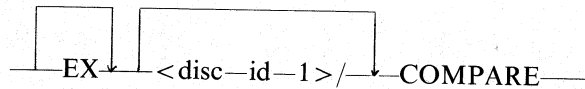


CO

CO (Compilation Utility)

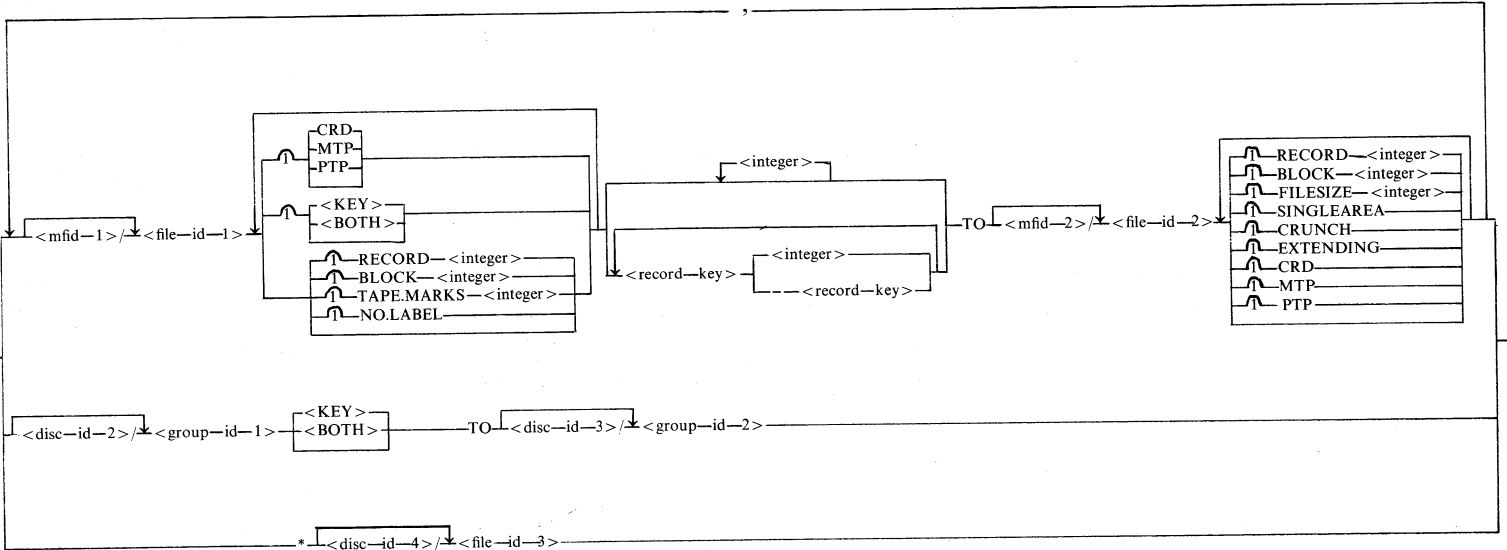


COMPARE

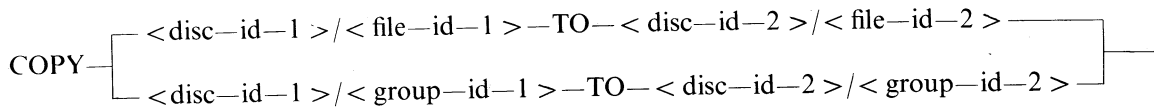


COPY (File Copy)

EX <disc-id-1>/ COPY

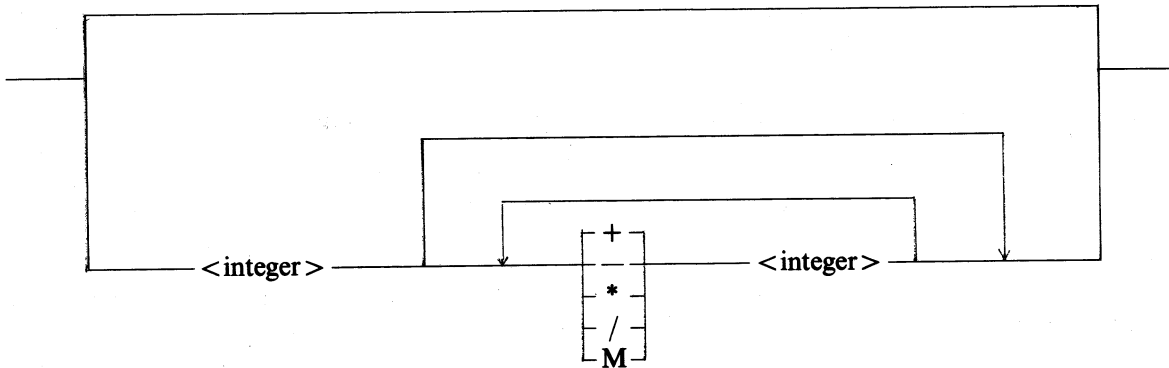


COPY (Stand-alone Disc Copy)



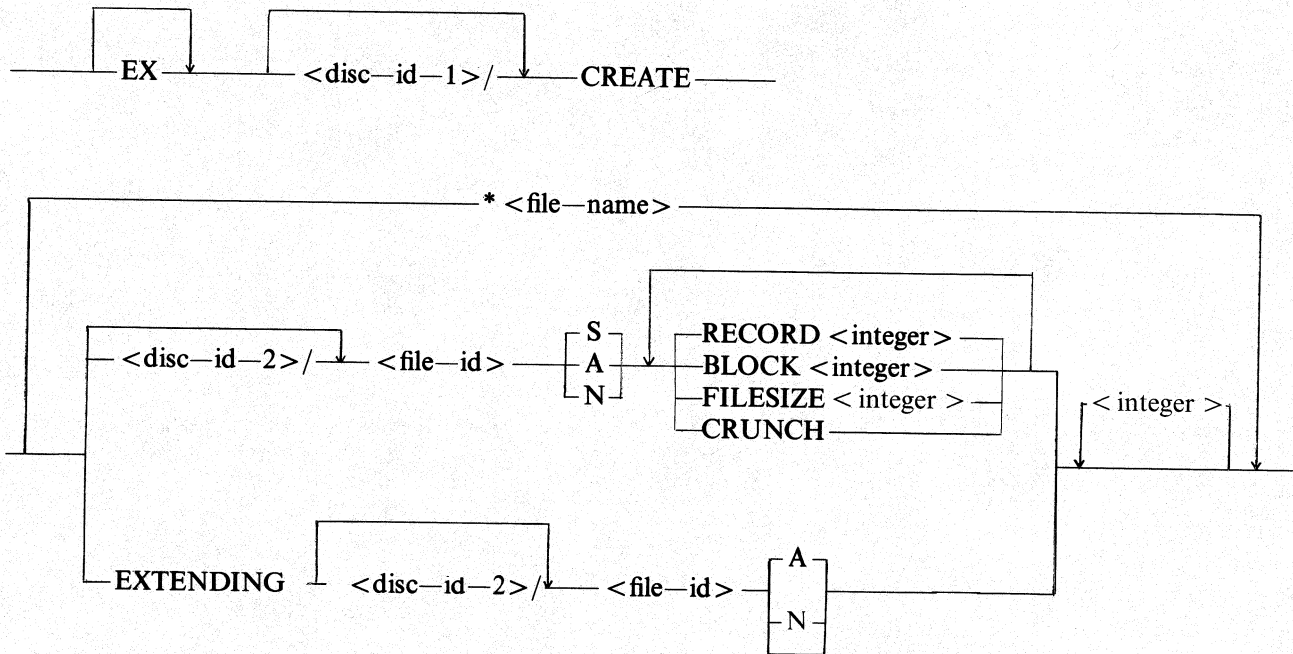
CP (Compute)

EX <disc-id>/ CP



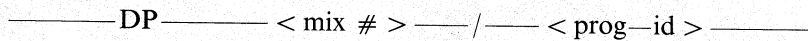
CREATE

CREATE (Disc File Create)



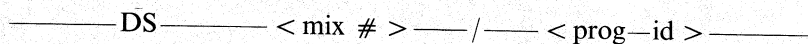
DP

DP (Discontinue and Dump)



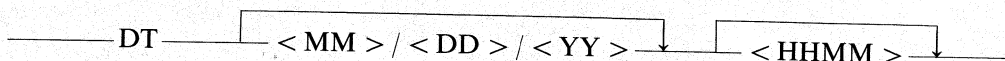
DS

DS (Discontinue Program)



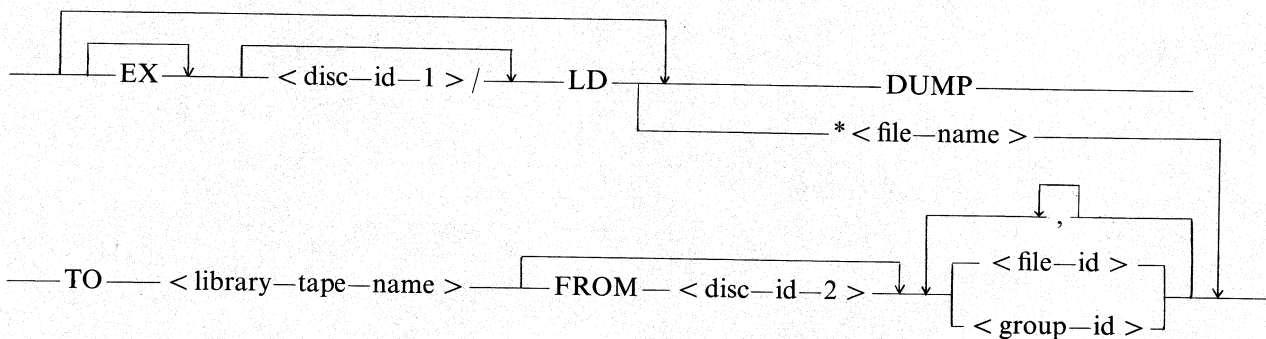
DT

DT (Date and Time)



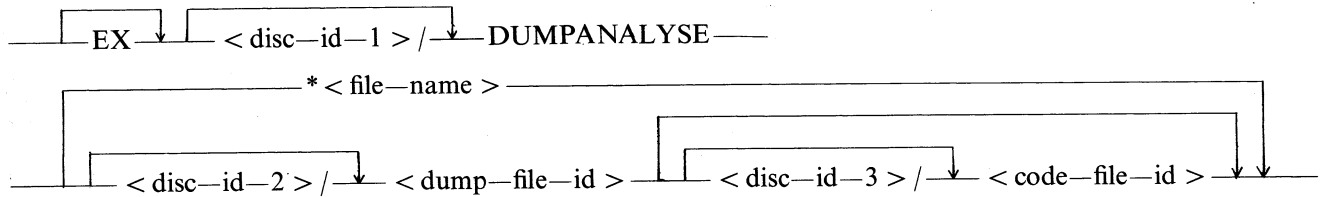
DUMP

DUMP (Dump Files to Library Tape)



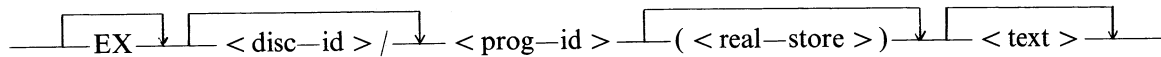
DUMPANALYSE

DUMPANALYSE (Analyse B80 Dump Files)



EX

EX (Execute)



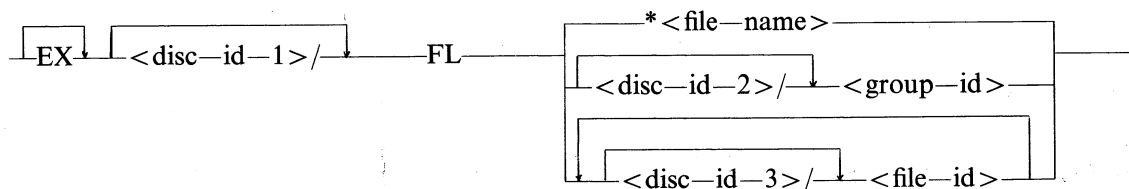
FE

FE (Initialise MTR Disc)



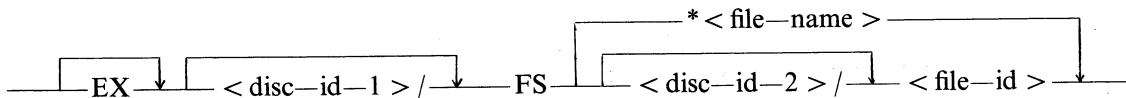
FL

FL (Display File Attributes on Self Scan)



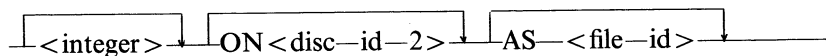
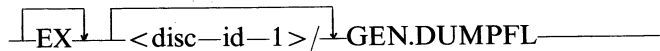
FS

FS (File Squash)



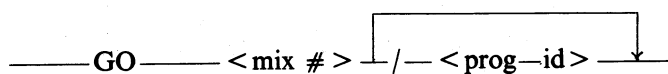
GEN.DUMPFL

GEN. DUMPFL (Create Empty Memory Dump File)



GO

GO (Restart a previously stopped program)



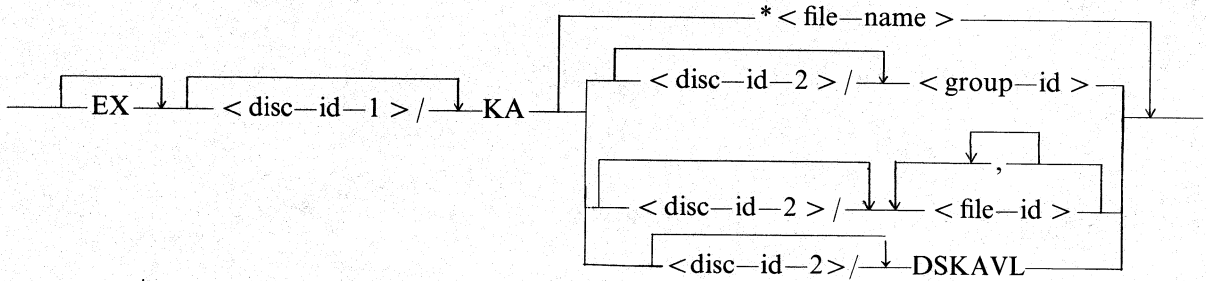
IN

IN (Initialise Disc)

—— IN ——

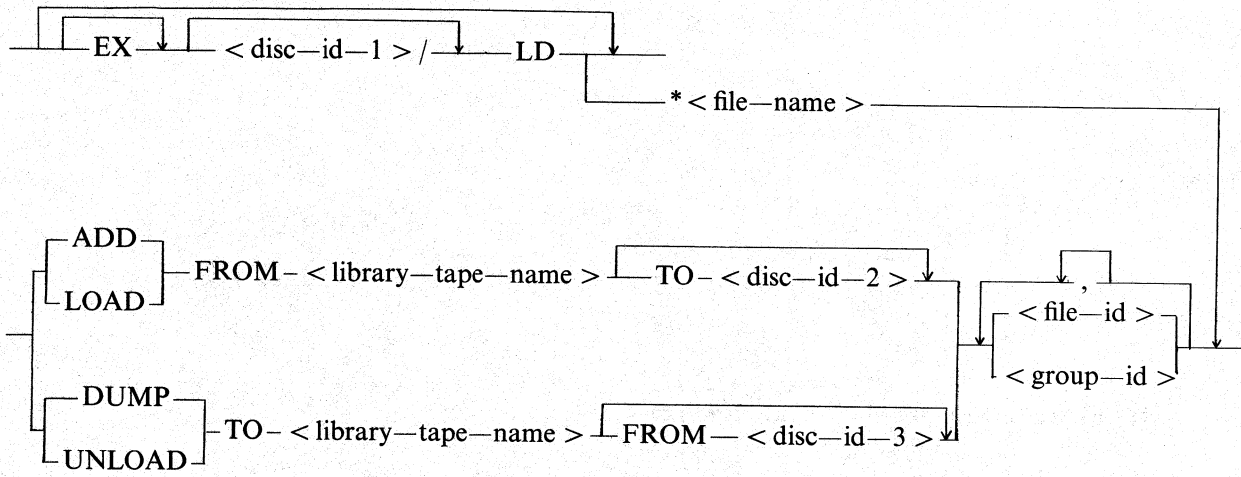
KA

KA (Analyse Disc Space Assignment)



LD

LD (Tape Library Utility)



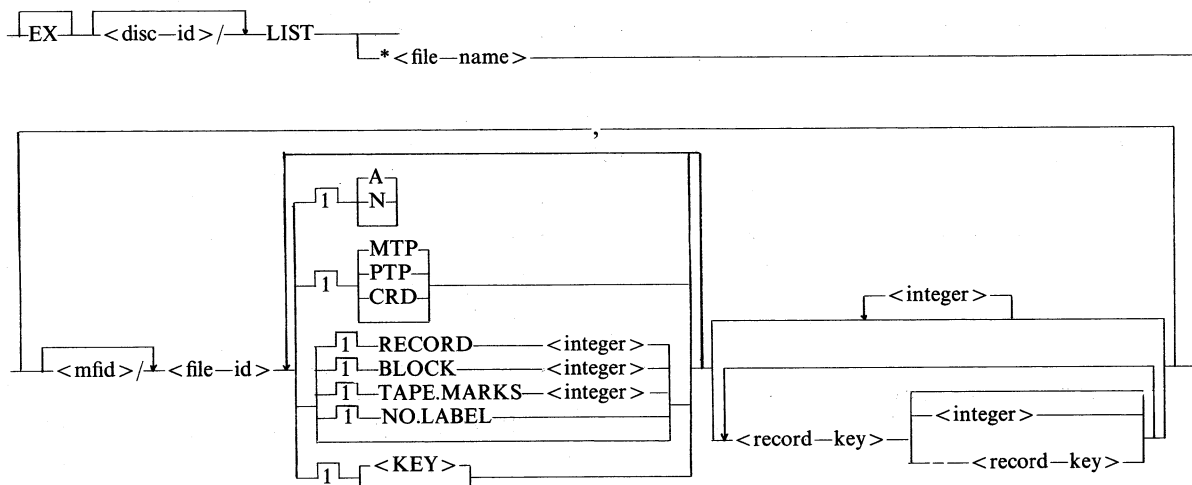
LD

LD (Load Disc)

—— LD - <disc-id > - FROM - <mfid > ——

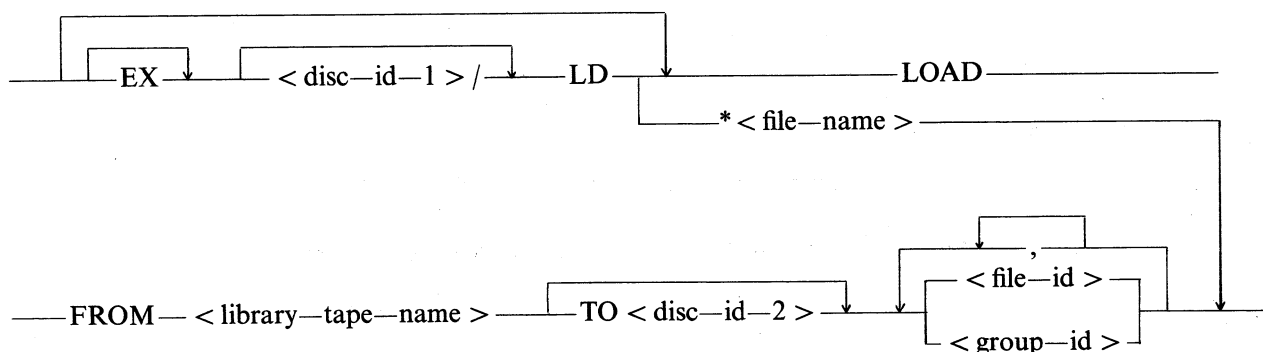
LIST

LIST (File List)



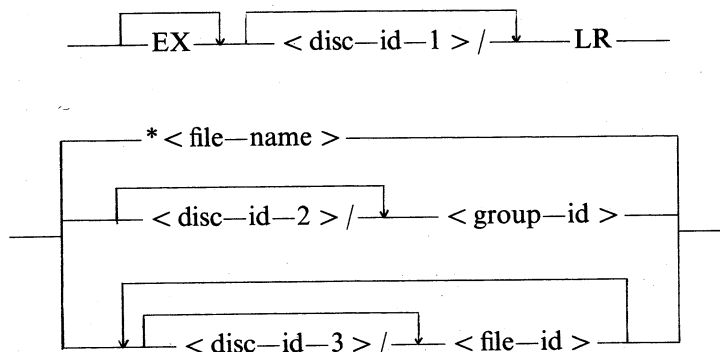
LOAD

LOAD (Load Library Tape Files)



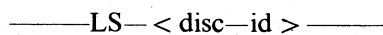
LR

LR (List Directory)



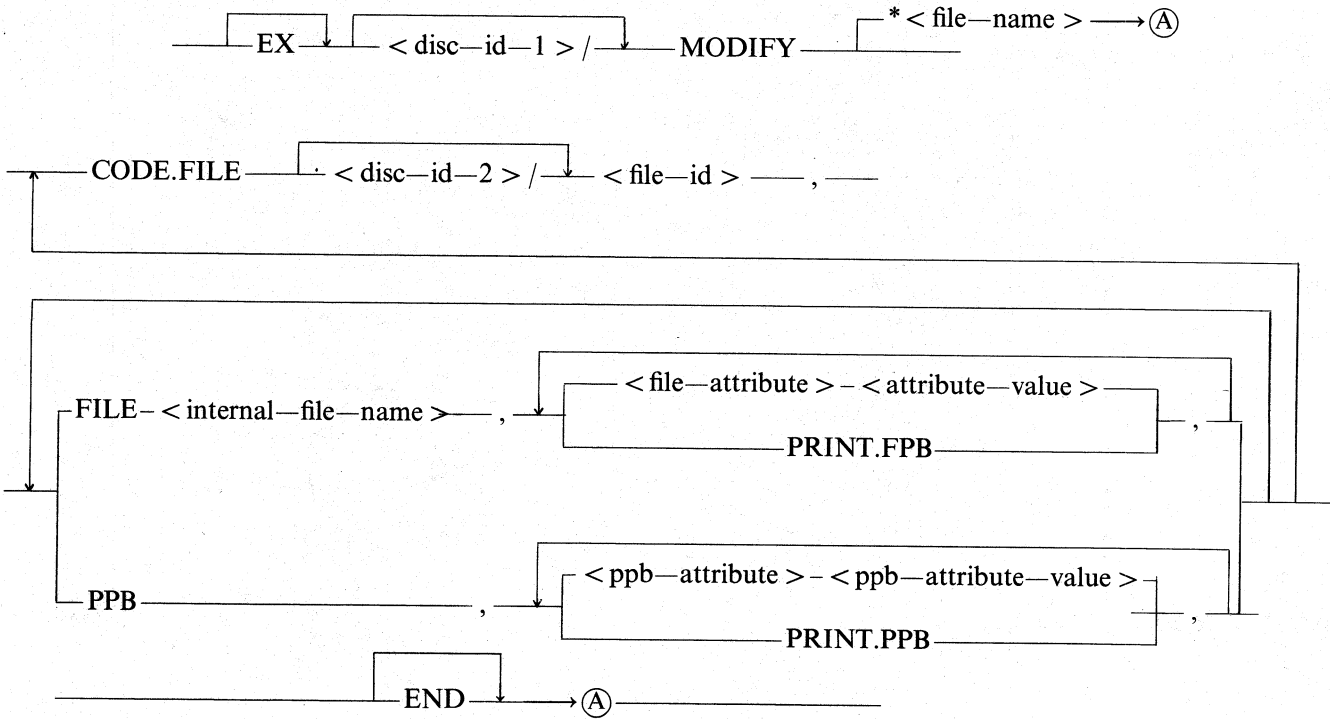
LS

LS (List Size)



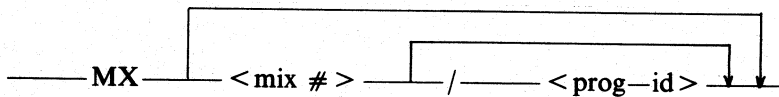
MODIFY

MODIFY (Program File Modification)



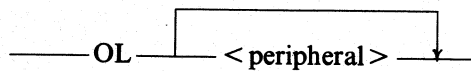
MX

MX (Diagnose Current Mix)



OL

OL (Request for status information of a peripheral)



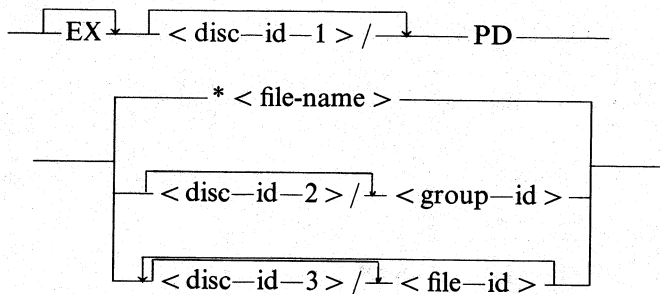
OL

OL (Print Status of Cassette Drives)



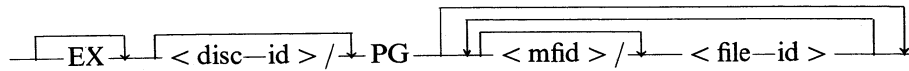
PD

PD (Interrogate Disc Directory)



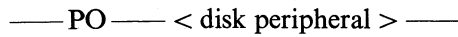
PG (Purge Tape)

PG



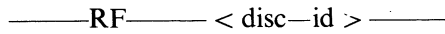
PO (Power off a disk drive)

PO



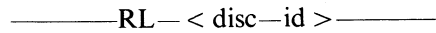
RF (Reformat Disc)

RF



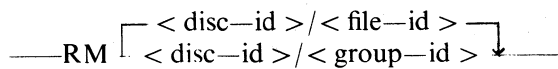
RL (Relabel Disc)

RL



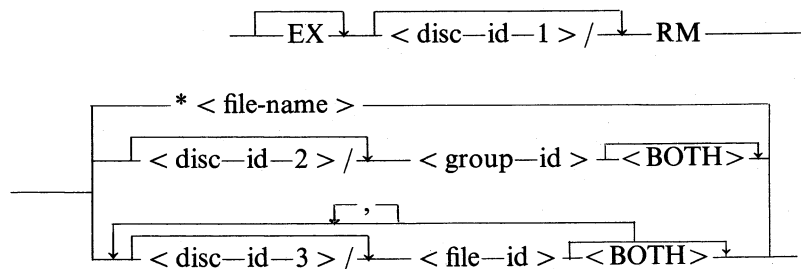
RM (Remove Files)

RM



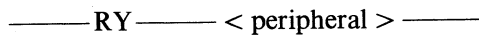
RM (Remove Disc Files from Directory)

RM



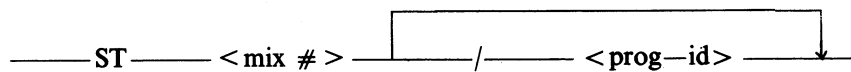
RY (Ready a peripheral)

RY



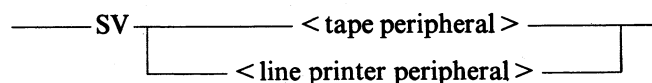
ST (Temporarily Suspend a Running Task)

ST



SV (Save Peripheral)

SV



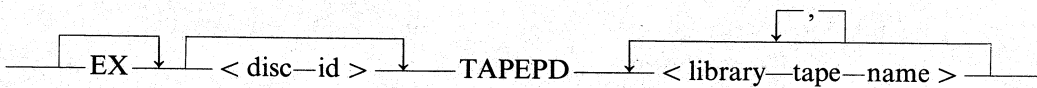
TAPELR (List Library Tape Directory)

TAPELR



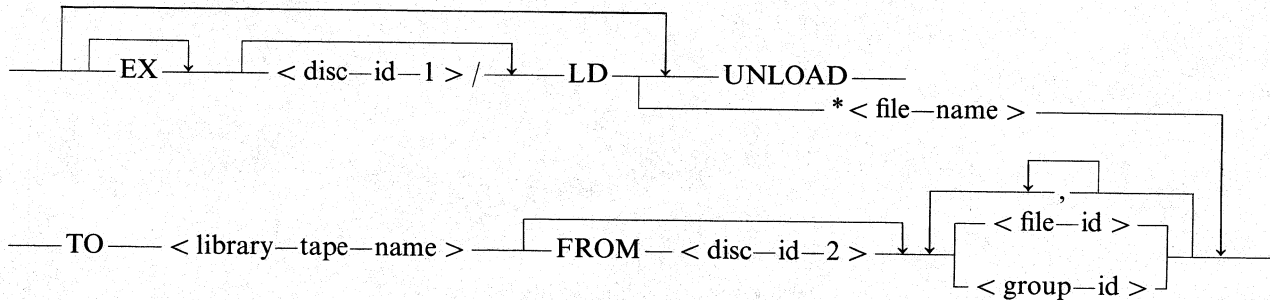
TAPEPD

TAPEPD (Interrogate Library Tape Directory)



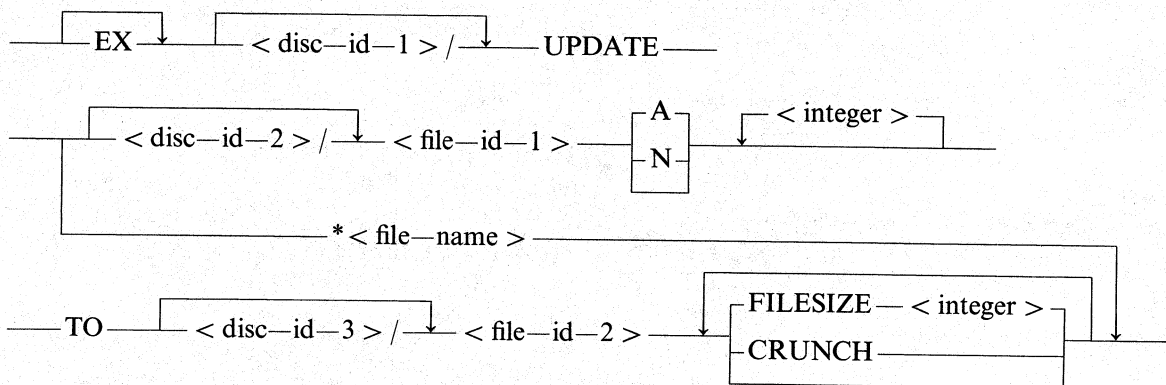
UNLOAD

UNLOAD (Unload files to Library Tape)



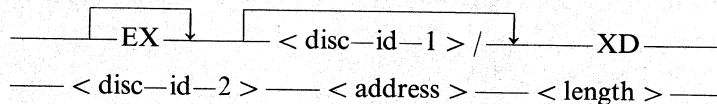
UPDATE

UPDATE (Disc File Update)



XD

XD (Delete Bad Sectors)



WS

WS (Warm Start)



APPENDIX B

SYSTEM OUTPUT MESSAGES

The following lists show the messages which may be displayed upon the system journal (SPO) by the Master Control Program (MCP) or the system utilities.

MCP OUTPUT MESSAGES

Two forms of messages are output by the MCP. The first form of message is considered to be self-documenting and no event number is included. These messages are listed with minimum explanation. The second form of message includes an event number enclosed in square brackets (for example [14]). These messages are listed in order of event number, and contain explanatory detail.

Basic Messages

NO CLOCK
INVALID DATE
INVALID TIME
INVALID PROGRAM ID
INVALID MIX
NULL MIX
< intrinsic—mnemonic > < peripheral > INVALID
(for example PO CSA INVALID)
< intrinsic—mnemonic > < peripheral > NOT ON SYSTEM
(for example SV LPB NOT ON SYSTEM)
< peripheral > IN USE
< peripheral > OK
(response to PO or SV)
PO ERROR ON < peripheral >
< mix >/< prog—id > ACPT
(request for AX response)
< mix >/< prog—id > DS'ED
< mix >/< prog—id > DP'ED
< mix >/< prog—id > STOPPED
< mix >/< prog—id > NOT STOPPED
(response to invalid GO)
< mix >/< prog—id > DS OR DP
(an unrecoverable condition has been encountered)
< mix >/< prog—id > ZIP: < text >
(The < text > is supplied by the ZIPPING program)
< mix >/< prog—id > DISP: < text >
(The < text > is supplied by the program)
< mix >/< prog—id > < intrinsic—mnemonic > INVALID
(for example IO/PD AD INVALID)
< peripheral > INVALID CONTROL CARD
< peripheral > INVALID CARTRIDGE

< mix > / < prog—id > BOJ PR IS < priority—letter >

< mix > / < prog—id > EOJ

< disc—id > / DMFIL < mix > CREATED

(This message records the creation of a dumpfile)

< mix > / < prog—id > DUMP FILE NOT CREATED

< faulty—input—message > INVALID

DS < mix > INVALID—NEEDS PROGRAM ID

CANNOT POWER DOWN SYSTEM. MIX NOT EMPTY.

< peripheral > REMOVED WITHOUT PO

Dxy < disc—id > SHOULD BE RE-INITIALISED SOON

Dxy < disc—id > BAD DISK—CANNOT BE LOADED

(The previous two messages are output by the “End of Life” detection which is performed during Automatic Volume Recognition (AVR) when a Burroughs Super Mini-disc is RY-ed.

Numbered Events

Exception Events 1–9, Software Information

FORMAT OF MESSAGE:

< MIX > / < PROGRAM—ID > [< EVENT # > < FILENAME > < PERIPHERAL > < FUNCTION >
< MESSAGE > ERROR WHILE IN < VERB > < STATUS > .

Event Message and Explanation

1 TAPE FORMAT

The ending label on a labeled tape is missing or invalid. The program has indicated it can handle this error itself.

2 PARITY

A hard parity error has been discovered on the device. The program has indicated it can handle such an error itself. < STATUS > shows the disk address for disk errors.

3 TIMEOUT

If the device is tape, then no data has been encountered for quite a distance on the tape (exact length depends upon the unit). For disk it means the cylinder specified could not be found. The program has indicated that it can handle such an error itself. < STATUS > shows the disk address for disk errors.

4 ADDRESS

The sector could not be found on disk (cylinder was found). Either the software has specified an invalid sector or the sector address on disk has become corrupt. The program has indicated it can handle such an error. < STATUS > shows the disk address.

5 REWIND

6 DESCRIPTOR

This message identifies all I/O errors that are not PARITY, TIMEOUT, ADDRESS, or REWIND errors. The program has indicated that it can handle such an error.

7 NON-FILE FULL ON MFID

It is possible for all entries in the non-file directory (available table) to be in use. If a file then releases its disk space it may not be possible to enter it in the non-file directory.

Exception Events 10–19, Software Suspensions

FORMAT OF MESSAGE:

< MIX > / < PROGRAM—ID > [EVENT #] [WAITING < FILENAME > < DEVICE >] [< MESSAGE >].

Event Message and Explanation

10 NO FILE

The MFID or FID requested cannot be located at OPEN time. If it is the MFID which cannot be located, the < FILENAME > will contain < MFID > /. If the FID cannot be located the < FILENAME > will contain the < FID >, if a single file device, or < MFID > / < FID > if a multi-file device. The operating system will cancel the suspension when the file is made present.

- 11 **DUPLICATE FILE**
 Either OPEN or CLOSE has detected two or more files of the same name and type. Contents of the < FILENAME > is based on the same criteria as no file (see event 11).
 The operating system will cancel the suspension when the duplicate is removed.
- 12 **NO USER DISK**
 The mentioned file requires an area of disk for its data but the specified disk is either too fragmented or too full to accommodate. The operator may remove files from the specified disk, or terminate the task. If the first option is used the operator must index a "GO" to the task to alert the operating system to try again. The < FILENAME > field will contain < MFID >/< FID >.
- 13 **DIRECTORY FULL**
 The file directory of the specified disk has no blank entries for creation of a new file. The < FILENAME > will contain < MFID >/< FID >. The system operator may remove files from that disk or terminate the task.
- 14 **DEVICE REQUIRED**
 An output device of the specified kind is needed. The operating system will cancel the suspension when the device becomes available. The < FILENAME > will be < FID > or < MFID >/< FID >.
- 15 **FORMS REQUIRED**
 The specified file has the FPB special forms flag set. This suspension notifies the system operator to insert the correct forms and then index an "AD" to the task < FILENAME > will be < FID >.
- 16 **BAD FILE TYPE**
 The specified file has been located, but its file type in the Disk File Header does not match that in the program's File Parameter Block. (Only applies to disk files.) The operator may remove the file presently on the disk and load the correct one or terminate the task. The operating system will cancel the suspension when the correct file is made present.
- 17 **NO FILE**
 The MFID requested cannot be located at open time. The < FILENAME > field will contain < MFID >. The operating system will cancel the suspension when the file is made present.
- 18 **DUPLICATE FILE**
 Either OPEN or CLOSE has detected two or more < MFID >s of the same type and name. The < FILENAME > field will contain < MFID >/. The operating system will cancel the suspension when the duplicate is removed.

Exception Events 20-40, Invalid Request on Class A or B Comm

FORMAT OF MESSAGE:

< MIX >/< PROGRAM-ID > [< EVENT # >] [CANNOT < VERB TEXT >] < PARAMETER >
 [< MESSAGE >].

Event Message and Explanation

****Resident Communicate Handler****

- 20 (NUL)
 The cause of the invalid request is an illegal verb.
- 21 NOT FIB
 Byte 1 of Class A or Class B communicate should contain a Data Segment Table Index of a File Information Block. This error is reported if that DST entry does not describe a FIB.
- 23 ATTRIBUTE MISMATCH
 The attributes of a file define which verbs may operate on that file. (Attributes include such things as the DEVICE, MYUSE, ACCESS MODE, etc.) Each Class A communicate is checked against the file's attributes. This message indicates a failure of this check, for example, READ on an output file, START on a non-disk file, etc.
- 24 BAD SEQUENCE
 This is a Class A error which may be noted when OPENED I/O with Sequential Access.
 1. A REWRITE was not immediately preceded by a successful READ.
 2. An OVERWRITE was immediately preceded by an OPEN or START communicate.
 3. An OVERWRITE or REWRITE was preceded by a conditional READ which failed.
- 25 BAD WORK AREA
 The Work Area segment specified within the FIB cannot be used, because either it indicates a FIB segment or it is a Read Only segment—yet the communicate requests a data transfer to that segment.

- 26 **ILLEGAL KEY**
The key requested on a Class A communicate for a disk file was equal to zero.
Non-resident Communicate Handler
- 30 **ALREADY OPEN**
The communicate requested an OPEN on an already Open file.
- 31 **ALREADY CLOSED**
The communicate requested a CLOSE of an already Closed file.
- 32 **BAD ADVERB**
The adverb to OPEN was determined to be illegal for any of the following reasons:
1. MYUSE equal to zero.
2. MYUSE incompatible with device, e.g. I/O for line printer.
3. Access Mode Random for non-disk device.
4. Access Mode not equal to Sequential or Random (if Stream Files are not implemented in this CMS Implementation).
- 33 **BAD BLOCK OR RECORD SIZE**
The Record and/or Block size has been determined to be incompatible or illegal for any of the following reasons:
1. Buffer or Record length equal to zero for new disk or tape files.
2. Record length exceeds physical Block size.
3. Buffer length not an integer multiple of Record length.
4. FPB or DFH buffer length is not a multiple of 180 if an old file is opened with a specified Buffer length other than zero.
5. DFH Buffer length is not a multiple of FPB Buffer length if an old file is opened with a specified Buffer length other than zero.
- 34 **BAD FILE SIZE**
The maximum file size specified exceeds 1048560, or 65535 if a single area file.
- 35 **BAD NUMBER OF BUFFERS**
The number of Buffers specified exceeded 16.
- 36 **BAD DEVICE**
The device requested is not supported by CMS. The device code is illegal.
- 37 **BAD FILE TYPE**
1. An OPEN has been requested on the "SYSTEM" file by an unprivileged user program.
2. An unprivileged user program has requested a CLOSE with Lock or Purge of a file with a SYSTEM VMFILE, or Firmware File Type.
- 38 **PROTECTION ERROR**
A privileged user attempted to CLOSE with Lock or Purge a VMFILE or Firmware file while the file was still in use.
- 39 **BAD FILENAME**
OPEN or CLOSE has detected an illegal special character imbedded in the filename.
- 40 **BAD KEYSIZE**
During the OPEN of a new index file, the MCP has discovered in the File Parameter Block (FPB) a key length of zero or one that exceeds 28 characters.

Exception Events 41–49, Fatal Device Errors

FORMAT OF MESSAGE:

< MIX >/< PROGRAM-ID > [< MESSAGE > ERROR WHILE IN < VERB > < STATUS >.

Event Message and Explanation

44 **TAPE FORMAT**

The ending label on a labeled tape is missing or invalid. The program has not indicated it can handle such an error.

45 **PARITY**

A hard parity error has been discovered on the device. The program has not indicated that it can handle such an error. < STATUS > shows the disk address for disk errors.

- 46 **TIMEOUT**
 If the device is tape, then no data has been encountered for quite a distance on the tape (exact length depends upon the unit). For disk it means the cylinder specified could not be found. < STATUS > shows the disk address for disk errors. In either case the program has not indicated that it can handle such an error.
- 47 **ADDRESS**
 The sector could not be found on disk (cylinder was found). Either the software has specified an invalid sector or the sector address on disk has become corrupt. The program has not indicated it can handle such an error. < STATUS > shows the disk address.
- 48 **REWIND**
- 49 **DESCRIPTOR**
 This error message identifies all I/O errors that are not PARITY, TIMEOUT, ADDRESS, or REWIND errors. The program has not indicated it can handle such an error.

Exception Events 50–69, Loader Detected Failures

FORMAT OF MESSAGE:

[< EVENT # >] LOAD FAILURE [< MESSAGE >].

- | Event | Message and Explanation |
|-------|---|
| 50 | DISK NOT FOUND
The specified disk is not online or not ready. |
| 51 | PROGRAM NOT FOUND
The specified file is not on disk. |
| 52 | MIX FULL
The requested program's Priority Class is full or the program presently executing requires that no other tasks may co-exist. |
| 53 | NO USER DISK
There is insufficient available space on the disk containing the program for the program's VMFILE. |
| 54 | INTERPRETER NOT FOUND
The interpreter file required cannot be located. |
| 55 | USER COUNT ERROR
This message occurs when the eighth user tries to use the program. |
| 56 | CODE FILE ERROR
The loader has noted an object code inconsistency. For example, an inconsistency exists if the PPB specifies an initiating message segment that is a Read Only segment. |
| 57 | INVALID LOAD REQUEST
There is an error in a parameter of the SCL load request, e.g. too many characters in the program name. |
| 58 | INSUFFICIENT MEMORY
There is not enough memory to hold this program's TCB and PCB. |
| 59 | MCS ALREADY PRESENT
Only one MCS may be in the mix. |
| 60 | DUPLICATE DISK
The DISK—ID specified in the load request has been found to be the name of two or more disks on the system. |
| 61 | NULL MIX REQUIRED
The program specified may only be loaded if the system is in a "NO MIX" state. The load is aborted if this condition is not met. |

Exception Events 70–99, Run Structure Problems

FORMAT OF MESSAGE:

< MIX >/< PROGRAM—ID > [< EVENT # >][SLICE < # > < MESSAGE >].

- | Event | Message and Explanation |
|-------|---|
| 70 | SEGMENT OUT OF RANGE
The segment number exceeds the number of segments declared in the program. |

- 71 SEGMENT SIZE ERROR
The Offset + Length value exceeds the declared size of the segment.
- 72 STACK OVERFLOW
The amount of Control Stack requested during execution has exceeded the declared stack size.
- 73 STACK UNDERFLOW
The designated code has attempted to retrieve more information from the Control Stack than is present.

Exception Events 100–169, Interpreters

FORMAT OF MESSAGE:

< MIX >/< PROGRAM-ID > [< EVENT # >] [< SOURCE REFERENCE > SEGMENT < PROGRAM SEGMENT # > ADDRESS < PROGRAM COUNTER ADDRESS > [< MESSAGE >].

Event Message and Explanation

****All Interpreters****

- 100 COMMUNICATE ERROR
The MCP has returned @80@ in Byte 0 of the Fetch Message on a communicate.
- 101 COMMUNICATE EOF ERROR
The MCP has returned an End of File indication in the Fetch Message (@20 20 00@) and the user has not specified any action to take if EOF occurs.
- 102 COMMUNICATE I/O ERROR
The MCP has returned an I/O error indication in the Fetch Message (@20 30 00@) and the user has not specified any action to take if an error occurs.
- 103 SEGMENT NUMBER ERROR
The interpreter has detected an invalid code or data segment number.
- 104 WRITE ERROR
The interpreter has detected an attempt to WRITE into a Read Only segment or literal.
The Code File has become corrupt or an error exists in the compiler or interpreter.
- 105 SEGMENT BOUNDARY VIOLATION
The interpreter, in calculating an address, has discovered that the address of the data or code is out of range.
The code file has become corrupt or an error exists in the compiler or interpreter.

****Interpreter for MPLII****

- 110 INVALID OP
The code file has become corrupt or an error exists in the MPLII compiler or interpreter.
- 115 DESCRIPTOR ACCESS ERROR
- 116 SEGMENT SIZE ERROR
- 117 ADDRESS ERROR
- 118 MESSAGE REFERENCE ERROR
- 119 STRING STORAGE ERROR
- 120 REMAP ERROR
- 121 SUBSTRING ERROR
- 122 INDEX ERROR
- 123 EXIT ERROR
- 124 CPA ERROR
- 125 DIVIDE ERROR
- 126 ZIP ERROR
- 127 BIT DESCRIPTOR ERROR
- 128 FPB ERROR
- 129 CONTROL STACK ERROR
- 130 DATA STACK ERROR
- 131 DECLARATION MODE ERROR
- 132 DATA STRUCTUR ERROR
- 136 BOUND ERROR
- 137 ARRAY BOUND ERROR
- 138 DO LOOP ERROR

- 139 CASE ERROR
 Interpreter for COBOL and RPG
- 140 CODE FILE ERROR
 Invalid S—Op—Code.
 The code file has become corrupt or an error exists in the COBOL/RPG compiler or interpreter.
- 141 CODE FILE ERROR
 Invalid COPX—greater than size of COP table. The code file has become corrupt or an error exists in the COBOL/RPG compiler or interpreter.
- 142 CODE FILE ERROR
 Alphanumeric field type not 8-bit unsigned. The code file has become corrupt or an error exists in the COBOL/RPG compiler or interpreter.
- 143 CODE FILE ERROR
 Invalid EDIT Micro Operator. The code file has become corrupt or an error exists in the COBOL/RPG compiler or interpreter.
- 144 CODE FILE ERROR
 Inline EDIT MASK not correctly terminated. The code file has become corrupt or an error exists in the COBOL/RPG compiler or interpreter.
- 145 CODE FILE ERROR
 EXAMINE source field error. The code file has become corrupt or an error exists in the COBOL/RPG compiler or interpreter.
- 146 CODE FILE ERROR
 EXAMINE parameter field not 8-bit unsigned one character. The code file has become corrupt or an error exists in the COBOL/RPG compiler or interpreter.
- 147 CODE FILE ERROR
 EXAMINE control byte error. The code file has become corrupt or an error exists in the COBOL/RPG compiler or interpreter.
- 148 CODE FILE ERROR
 COMPARE for CLASS—CLASS and FIELD type incompatible. The code file has become corrupt or an error exists in the COBOL/RPG compiler or interpreter.
- 149 SUBSCRIPTED OR INDEXED SUBSCRIPT OR INDEX
 The code file has become corrupt or an error exists in the COBOL/RPG compiler or interpreter.
- 160 PERFORM STACK OVERFLOW
 Indicates too many PERFORMS without a return for the Perform Stack specified at compile time (if not specified then the default was used). If this did not result from a programming error, the Perform Stack should be increased.
- 161 NON-POSITIVE SUBSCRIPT
 Subscripts must be positive.
- 162 ARRAY BOUND VIOLATION
 Subscript outside upper bound of OCCURS Clause.
- 164 TRANSLATION SOURCE ERROR
- 166 INVALID SIGN CODE

Exception Events 170–199, Sort

FORMAT OF MESSAGE:

< MIX >/< PROGRAM > [< EVENT # >] < MESSAGE >. Note that some CMS implementations of SORT may not display the event number.

Event Message and Explanation

- 170 DUPLICATE RECORD < RECORD NUMBER >
 Only for keyfile creation. Another Record in the file has the same key as this record.
- 171 ILLEGAL INDEX KEY IN RECORD < RECORD NUMBER >
 Only for keyfile creation. Either the key field is all binary 0 or has one or more bytes with hex 'FF'. This record will not be referenced from the keyfile.
- 172 RECORDS LOST OR GAINED BY SORT—MERGE
 Probably indicates an error in SORTINTRINS. See note below for what to do.

- 173 < NUMBER > DUPLICATE RECORDS
Tells the operator the total number of records that have duplicates. See event 170.
- 174 < NUMBER > RECORDS CONTAINING INVALID INDEX KEYS
Tells the operator the total number of records that contain invalid index keys. See event 171 for further information.
- 175 < NUMBER > RECORDS DELETED
Records with hex FF in every byte position will be deleted by SORT. This message informs the operator how many records were deleted.
- 176 < NUMBER > RECORDS MERGED
For MERGE only. Informs the operator of the total number of records merged from all files.
- 177 < NUMBER > FILES MERGED
For MERGE only. Informs operator of the total number of files merged. Should be the same as the number of files requested in the Sort Specs.
- 178 SORT-MERGE OUTPUT FILE NOT CREATED
This message occurs if SORTINTRINS is DS-ed.
- 179 SORT-MERGE ABNORMAL EOJ
Informs the operator of an early termination due to errors.
- 180 SORT-MERGE SOFTWARE ERROR
Error in SORTINTRINS. See note below for what to do.
- 181 < NUMBER > RECORDS REFERENCED BY KEYFILE/TAGFILE
Only for keyfile/tagfile creation. Tells the operator the number of entries in the keyfile/tagfile.
- 182 NO INITIATING MESSAGE
The SORT INTRINSIC requires a properly coded Initiating Message. This should be properly formatted by SORT or Sorts within programming languages such as COBOL. This message probably indicates an attempt to execute the SORT INTRINSIC directly.
- 183 < NUMBER > RECORDS SORTED
This is a normal message to inform the operator how many records have been Sorted.
- 184 FILE ERROR (< NUMBER >) ON SORT-MERGE FILE < FILENAME >
The < NUMBER > means:
1. EOF on Output File.
 2. PARITY on Input File.
 3. EOF on Sort Workfile.
 4. Bad Disk Address.
 5. SORT Workfile Error.
 6. Input File Error.
 7. Output File Error.
- Except for 2, this probably indicates an error in SORTINTRINS—in which case see note below for what to do.
- 185 UNORDERED MERGE INPUT FILE < FILENAME >
The files to be merged were found not to be in increasing/decreasing key value. Either the file is incorrect or the key position has been incorrectly specified.
- 186 TOO MANY RECORDS FOR SORT-MERGE
A machine dependent limitation. For the B80 this limit is currently about 65000 records.
- 187 DUPLICATE RECORDS—KEYFILE NOT BUILT
If NO DUPLICATES are specified and duplicates exist, the keyfile will not be built and this message will be given.
- 188 INIT MESSAGE INVALID
The initiating message to the SORT INTRINSIC is not in the proper format. This could possibly be caused by a fault in the program that zipped the SORT INTRINSIC.
- 189 SORT—MERGE INITIATED FROM < MIX >/< PROGRAM—10 >
This is a normal event to inform the operator which program called the SORT INTRINSIC.
- Note: All SORT errors should be accompanied by either the Sort Spec or the COBOL program that invoked the SORT INTRINSICS as well as the input file(s). Since SORTINTRINS is a machine dependent implementation the method of getting a program dump may vary. To get a program dump on the B80:
Rerun with GT on. This will cause SORTINTRINS to dump its run structure on the console printer.
Printing will take about 15 minutes.

This note is for the assistance of Burroughs Field Support Personnel. The normal user should notify Burroughs Field Support in the event of any of these errors.

UTILITY OUTPUT MESSAGES

The following messages are output by the utility shown in brackets below each message. Note that the Sort Language Processor messages may be output to line printer in some cases. All messages which commence with a variable (for example "< file-id >") are listed first, followed by an alphabetic list of the remainder.

< disc-id > NOT ON LINE
(COPY)

< file-id > AND < file-id > REMOVED
(RM)

< file-id > CHANGED TO < file-id >
(CH)

< file-id > DUMPED
(LD)

< file-id > EXHAUSTED DURING < integer >
< integer >
(COPY, LIST)

< file-id > EXHAUSTED DURING RANGE
< key-1 > { -< key-2 > }
 { < integer > }
(COPY, LIST)

< file-id > FILE IDENTIFIER TOO LONG
(CH)

< file-id > IS A SYSTEM FILE
(RM)

< file-id > LOAD/DUMP DISCREPANCY
(LD)

< file-id > LOADED
(LD)

< file-id > NO LONGER ON LINE
(LD)

< file-id > NOT ACCEPTABLE—RECORD SIZE
OF < integer > IS GREATER THAN THE
MAXIMUM SPECIFIED FOR THIS RUN—
RESUBMIT
(LIST)

< file-id > NOT CHANGED—ILLEGAL REQUEST
(CH)

< file-id > NOT CHANGED—NOT ON LINE
(CH)

< file-id > NOT CHANGED—IN USE
(CH)

< file-id > NOT CHANGED — < file-id >
ALREADY ON DISK
(CH)

< file-id > NOT COPIED—ILLEGAL REQUEST
(COPY)

< file-id > NOT DUMPED—
{ HAS BEEN REMOVED }
{ IN OUTPUT USE }
{ HAS BEEN ALTERED }

(LD)

< file-id > NOT FOUND
(any utility)

< file-id > NOT LOADED—ALREADY ON DISK
(LD)

< file-id > NOT ON LINE
(FS)

< file—id > NOT REMOVED—IN USE
(RM)

< file—id > NOT REMOVED—NOT FOUND
(RM)

< file—id > NOT REMOVED OR DUMPED—
IN USE
(LD)

< file—id > NOT REMOVED—SYSTEM FILE
(RM)

< file—id > REMOVED
(LD, RM)

< file—id > REQUIRES OVERFLOW DISK
< disc—id >
(PD)

< file—id > SQUASHED FROM < integer >
RECORDS TO < integer > RECORDS
(FS)

< file—id > TO < file—id > BAD ATTRIBUTES
(COPY)

< file—id > TO < file—id > COPY COMPLETED
(COPY)

< file—id > TO < file—id > COPY DISCREPANCY
(COPY)

< group—id > ON < disc—id >
CONTAINS: — < file—list >
(PD)

< identifier > FILE IDENTIFIER TOO LONG
(CH)

< integer > < integer > < file—id > NOT COPIED
(COPY)

< integer > < integer > IN < file—id >
NOT LISTED
(LIST)

@ < length > @ SECTORS FROM @ < address > @
DELETED
(XD)

< mfid > HAS BEEN PURGED
(PG)

< mfid > INVALID DIRECTORY ON TAPE
(CHECKADUMP)

< mfid > NOT A RECOGNISED DUMP TAPE
(LD, CHECKADUMP, TAPELR, TAPEPD)

< mfid > NOT ON LINE
(PG)

< mfid > NO WRITE PERMIT
(PG)

< parameter—list > ILLEGAL PARAMETER LIST
(CHECKADUMP, XD)

ALPHANUMERIC KEY LENGTH NOT EVEN
NUMBER OF 4-BITS NEAR COL XXX
(SORT LANGUAGE PROCESSOR)

ALTHOUGH WITH DIFFERENT ATTRIBUTES
(LD)

ATTRIBUTE VALUE MISSING
(MODIFY)

ATTRIBUTE—VAL INCONSISTENT
(MODIFY)

AVAILABLE TABLE FULL—ENTRY
@ < address > @ @ < length > @ LOST
(XD)

< BLOCK FACTOR > OF 0 NOT ALLOWED,
1 ASSUMED NEAR COL XXX
(SORT LANGUAGE PROCESSOR)

< BLOCK FACTOR > TOO LARGE,
MAXIMUM ASSUMED NEAR COL XXX
(SORT LANGUAGE PROCESSOR)

BUFFER SIZE TOO BIG FOR < IN/OUT MEDIA >
NEAR COL XXX
(SORT LANGUAGE PROCESSOR)

BUFFER SIZE TOO LARGE NEAR COL XXX
(SORT LANGUAGE PROCESSOR)

< CHARACTER STRING > EXPECTED NEAR
COL XXX
(SORT LANGUAGE PROCESSOR)

CODE FILE NAME IN ERROR
(MODIFY)

COMPARISON ERROR ON < file-id >
AROUND END OF FILE
(CHECKADUMP)

COMPARISON ERROR ON < file-id >
(AROUND RECORD < integer >)
(CHECKADUMP)

COMPARISON ERROR ON < file-id >
FILE NOT AVAILABLE FOR CHECK
(CHECKADUMP)

COMPARISON ERROR ON < file-id >
FILE NOT FOUND FOR CHECK
(CHECKADUMP)

COMPARISON ERROR ON < mfid >
ON DISK FILE HEADERS
(CHECKADUMP)

CP: DIVISION BY ZERO
(CP)

CP: HEX. NO. WITH MISSING "@"
(CP)

CP: < hex-value > = < decimal-value >
(CP)

CP: INVALID OPERATOR
(CP)

CP: MISSING OPERAND
(CP)

CP: NUMBER TOO LARGE
(CP)

CP: OVERFLOW
(CP)

(CURRENT SUM OF) KEY LENGTH(S) OUT OF
RANGE NEAR COL XXX
(SORT LANGUAGE PROCESSOR)

DEVICE—MYUSE INCONSISTENT
(MODIFY)

< DIGIT STRING > EXPECTED NEAR COL XXX
(SORT LANGUAGE PROCESSOR)

DISCREPANCIES FOUND BETWEEN DUMP
TAPE < mfid > AND DISK < disc-id >
(CHECKADUMP)

DISK < disc-id > NOT OPENED—NOT ON LINE
(LR, KA, RM, CH, FL)

DISK < disc-id > NOT AVAILABLE
(LD)

DISK < disc-id > FOR XD NOT AVAILABLE
(XD)

DUPLICATE < IN—FILE PARAMS >,
LATEST INSTANCE NEAR COL XXX
(SORT LANGUAGE PROCESSOR)

< DUPLICATE OPTION > VALID IN
INDEX—KEYFILE SORT ONLY
(SORT LANGUAGE PROCESSOR)

ERROR WHEN EXPECTING \$
(CO)

EXPECTED BRACKET NOT FOUND,
“(” INSERTED NEAR COL XXX
(SORT LANGUAGE PROCESSOR)

EXPECTED BRACKET NOT FOUND,
“)” INSERTED NEAR COL XXX
(SORT LANGUAGE PROCESSOR)

EXPECTED SLASH NOT FOUND,
“/” INSERTED NEAR COL XXX
(SORT LANGUAGE PROCESSOR)

EXTRA DIGITS IN OVERLENGTH STRING
IGNORED NEAR COL XXX
(SORT LANGUAGE PROCESSOR)

EXTRA “FILE IN” NEAR COL XXX
(SORT LANGUAGE PROCESSOR)

EXTRA “KEY” NEAR COL XXX
(SORT LANGUAGE PROCESSOR)

FILE ID TOO LONG < file—id >
(DUMPANALYSE)

FILE NAME NOT FOUND
(MODIFY)

< FILE SIZE OPT > IGNORED SINCE OUT OF
RANGE NEAR COL XXX
(SORT LANGUAGE PROCESSOR)

FILE-SIZE TOO LARGE
(MODIFY)

< FILE STATEMENT > ALREADY PROCESSED,
NOW NEAR COL XXX
(SORT LANGUAGE PROCESSOR)

FILE TYPE IS NOT SOURCE OR DATA
(AMEND, CREATE, UPDATE)

FIRST UNIT NUMBERED 0 RATHER THAN 1
NEAR COL XXX
(SORT LANGUAGE PROCESSOR)

ILLEGAL PARAMETER LIST
(Any utility)

IGNORE < FILE SIZE OPT > SINCE LESS THAN
< PARTIT OPT >
(SORT LANGUAGE PROCESSOR)

ILLEGAL PARAMETER LIST—ATTRIBUTE
SPECIFICATION INVALID
(AMEND, CREATE, UPDATE)

ILLEGAL PARAMETER LIST—JOB
TERMINATED
(CO)

ILLEGAL PARAMETER LIST—TABS ERROR
(AMEND, CREATE, UPDATE)

ILLEGAL TO DELETE INPUT FILE,
< PURGE OPT > IGNORED
(SORT LANGUAGE PROCESSOR)

ILLEGAL TO OVERWRITE INPUT FILE WITH
TAG/KEY FILE
(SORT LANGUAGE PROCESSOR)

IN- & OUT-FILE RECORD SIZES MADE EQUAL
(SORT LANGUAGE PROCESSOR)

IN- & OUT-FILE RECORD SIZES MUST BE IDENTICAL
(SORT LANGUAGE PROCESSOR)

INCORRECT ATTRIBUTE
(MODIFY)

INDEX-KEYFILE KEY LENGTH NOT EVEN NUMBER OF 4-BITS
(SORT LANGUAGE PROCESSOR)

INDEX-KEYFILE SORT KEY MUST BE "... A UA/UN"
(SORT LANGUAGE PROCESSOR)

INDEX-KEYFILE SORT KEY MUST START ON BYTE BOUNDARY
(SORT LANGUAGE PROCESSOR)

INDEX-KEYFILE SORT KEY TOO LONG
(SORT LANGUAGE PROCESSOR)

INDEX PARAM MUST BE "OUT... (KEYFILE/TAGFILE)"
(SORT LANGUAGE PROCESSOR)

INDEX < SORT TYPE OPTION > NOT SPECIFIED
(SORT LANGUAGE PROCESSOR)

INPLACE INTRINSIC IGNORES < MEMORY OPTION >
(SORT LANGUAGE PROCESSOR)

INPLACE SORT MUST HAVE IDENTICAL IN- & OUT-FILES
(SORT LANGUAGE PROCESSOR)

INPUT FILES RECORD SIZES NOT IDENTICAL NEAR COL XXX
(SORT LANGUAGE PROCESSOR)

INPUT, OUTPUT FILES IDENTICAL, < PURGE OPT > INSERTED
(SORT LANGUAGE PROCESSOR)

INTERPRETER VERSION NOT SUPPORTED
< file-id >
(DUMPANALYSE)

INVALID ATTRIBUTE FOR OLD FILE
(CO)

INVALID CALL OF *FILE-NAME—
JOB TERMINATED
(CO)

INVALID CHARACTER IN IDENTIFIER
< identifier >
(any utility)

INVALID CHARACTERS IN FILE < file-id >
(DUMPANALYSE)

INVALID DEVICE TYPE < text >
(CO)

INVALID KEYWORD IN FILE MOD
(CO)

INVALID KEYWORD FOUND AFTER DOLLAR
(CO)

INVALID KEYWORD IN INPUT—_l < text >
(CO)

INVALID VALUE IN STACK CLAUSE—
DEFAULT USED
(CO)

KEYFILE < file-id > NOW POINTS TO DATA FILE < file-id >
(CH)

KEYFILE < file-id > RECONSTRUCTED
(FS)

KEYFILE SORT FAILURE
(FS)

KEY OVER-RUNS RECORD BOUNDARY
(SORT LANGUAGE PROCESSOR)

< KEY STATEMENTS > ALREADY PROCESSED,
NOW NEAR COL XXX
(SORT LANGUAGE PROCESSOR)

KEYWORD IN ERROR
(MODIFY)

KEYWORD ONLY VALID FOR FILE
MODIFICATION—IGNORED
(CO)

KEYWORD TOO LARGE IN INPUT
(CO)

< LETTER STRING > EXPECTED NEAR
COL XXX
(SORT LANGUAGE PROCESSOR)

< MEDIA > MUST BE DISK IF SORTING BACK
TO IN-FILE
(SORT LANGUAGE PROCESSOR)

MERGE INTRINSIC IGNORES
< FILE SIZE OPTION >
(SORT LANGUAGE PROCESSOR)

MERGE INTRINSIC IGNORES
< MEMORY OPTION >
(SORT LANGUAGE PROCESSOR)

MERGE INTRINSIC NEEDS AT LEAST 2 INPUT
FILES
(SORT LANGUAGE PROCESSOR)

MERGE < SORT TYPE OPTION >
NOT SPECIFIED
(SORT LANGUAGE PROCESSOR)

< M—FILE/DP ID > IGNORED ON
NON-MAGNETIC MEDIA FILE NEAR COL XXX
(SORT LANGUAGE PROCESSOR)

MIN LENGTH OF SN KEY IS TWO 4-BIT UNITS
NEAR COL XXX
(SORT LANGUAGE PROCESSOR)

MIN LENGTH OF SSA KEY IS FOUR 4-BIT UNITS
NEAR COL XXX
(SORT LANGUAGE PROCESSOR)

MISSING “<..>/<..>” OR “<..>”
NEAR COL XXX
(SORT LANGUAGE PROCESSOR)

MISSING “FILE IN” NEAR COL XXX
(SORT LANGUAGE PROCESSOR)

MISSING “KEY” NEAR COL XXX
(SORT LANGUAGE PROCESSOR)

MISSING SEPARATOR
(MODIFY)

MT < mfid > DUMPED ON < date > CONTAINS:
< file—list >
(TAPEPD)

NESTED MACRO CALL FOUND—
JOB TERMINATED
(CO)

NO COMPILER GIVEN BEFORE FILE—
JOB TERMINATED
(CO)

NO COMPILER SPECIFIED—JOB TERMINATED
(CO)

NO DISCREPANCIES BETWEEN DUMP TAPE
< mfid > AND DISK < disc—id >
(CHECKADUMP)

NO FILES FOUND FOR CHANGING IN THE
FAMILY < group—id >
(CH)

{ NO FILES IN THE FAMILY }
 { NO FILE }
 < identifier > ON { TAPE } { DISK } < identifier > }
 FOR { LOAD }
 { ADD }
 { DUMP }
 { UNLOAD }
 (LD)

NO FILES FOUND FOR REMOVAL IN THE
 FAMILY < group—id >
 (RM)

NO FILES FOUND IN THE FAMILY < group—id >
 (COPY, PD)

NO FILES TO DUMP
 (LD)

NO FILES TO LOAD
 (LD)

NO < FILE STATEMENT > SPECIFIED
 (SORT LANGUAGE PROCESSOR)

NO < KEY STATEMENT > SPECIFIED
 (SORT LANGUAGE PROCESSOR)

NO KEYWORD FOUND AFTER \$
 (CO)

NO OUTPUT GENERATED BY KA
 (KA)

NO RECORDS FOR LISTING FROM < file—id >
 (LIST)

NO SPECIFICATION GIVEN
 (all utilities)

NO SPECIFICATION GIVEN—JOB TERMINATED
 (CO)

NOT NECESSARY TO PURGE CARD FILE
 NEAR COL XXX
 (SORT LANGUAGE PROCESSOR)

NOT ON LINE < file—list >
 (PD)

NUMBER TOO BIG, MAXIMUM VALUE
 ALLOWABLE ASSUMED NEAR COL XXX
 (SORT LANGUAGE PROCESSOR)

ON LINE < file—list >
 (PD)

ONLY @ < length > @ SECTORS CAN BE
 DELETED
 (XD)

ONLY INDEX SORT CAN SPECIFY
 “KEYFILE/TAGFILE”
 (SORT LANGUAGE PROCESSOR)

ONLY ONE IN-FILE LEGAL FROM MULTIFILE
 TAPE NEAR COL XXX
 (SORT LANGUAGE PROCESSOR)

ONLY ONE KEY LEGAL IN INDEX-KEYFILE
 SORT
 (SORT LANGUAGE PROCESSOR)

OVERLENGTH PART OF < LABEL NAME >
 IGNORED NEAR COL XXX
 (SORT LANGUAGE PROCESSOR)

PACK < disc—id > NOT AVAILABLE
 (any utility)

PACK ID TOO LONG < disc—id >
 (DUMPANALYSE)

< PARTITION OPT > VALID FOR
 INPLACE/REGULAR SORT ONLY
 (SORT LANGUAGE PROCESSOR)

PARTITION SIZE TOO BIG, SORT TO EOF
ASSUMED NEAR COL XXX
(SORT LANGUAGE PROCESSOR)

PARTITN TO IN-FILE, ALTERED OUT-FILE
PARAMS IGNORED
(SORT LANGUAGE PROCESSOR)

PART OF < FILE STATEMENT > MISSING,
NOW NEAR COL XXX
(SORT LANGUAGE PROCESSOR)

REC. NOT INTEGRAL OF BUF.
(MODIFY)

< RECORD SIZE > OUT OF RANGE
NEAR COL XXX
(SORT LANGUAGE PROCESSOR)

REMAINDER OF STATEMENT MISSING
(SORT LANGUAGE PROCESSOR)

SECTORS FOR XD NOT IN AVAILABLE
TABLE
(XD)

< SEPARATOR STRING > EXPECTED NEAR
COL XXX
(SORT LANGUAGE PROCESSOR)

< SIGN POSITION > GIVEN FOR UNSIGNED
KEY NEAR COL XXX
(SORT LANGUAGE PROCESSOR)

SSA KEY MUST START ON BYTE BOUNDARY
NEAR COL XXX
(SORT LANGUAGE PROCESSOR)

STATEMENT PARTLY IGNORED,
ILLEGAL WORD NEAR COL XXX
(SORT LANGUAGE PROCESSOR)

TABLE SIZE EXCEEDED
(KA)

TOO MANY BUFFERS
(MODIFY)

TOO MANY FILE SPECIFICATIONS NEAR
COL XXX
(SORT LANGUAGE PROCESSOR)

TOO MANY KEY SPECIFICATIONS NEAR
COL XXX
(SORT LANGUAGE PROCESSOR)

UNLABELLED TAPE—HAS BEEN PURGED
(PG)

UNLABELLED TAPE—NO WRITE PERMIT
(PG)

UNSUPPORTED < IN/OUT MEDIA >
NEAR COL XXX
(SORT LANGUAGE PROCESSOR)

UNSUPPORTED < SORT TYPE OPTION >
NEAR COL XXX
(SORT LANGUAGE PROCESSOR)

< USER OPTION > ALREADY INVOKED,
LATEST USE NEAR COL XXX
(SORT LANGUAGE PROCESSOR)

WARNING BIL DUMPPFILE, SECOND
PARAMETER IN INIT. MESS IGNORED
(DUMPANALYSE).

WARNING—PROG—PACK—ID TRUNCATED
TO 7 CHARS
(CO)

WARNING—PROG—ID TRUNCATED TO
12 CHARS
(CO)

WARNING—THE FILE ATTRIBUTES OF
SOURCEOUT WERE INCOMPLETE
(CO)

ZIP FAILURE— < reason >
(CO)

Notes

